2004年度 修士論文

地理属性を考慮したZoneDHTによる分散コンテンツ管理モデル

提出日： 2005 年 2 月 2 日
指導： 村岡洋一教授

早稲田大学大学院 理工学研究科 情報・ネットワーク専攻
学籍番号： 3603U137-5

宮之前　謙一

目 次

図 目 次

表 目 次

第1章　序論

本章では、始めに本稿の全体の流れについて述べ、その後、本研究の背景、目的、実装と評価、想定サービスについて述べ、最後に本稿の構成を述べる。

1.1　はじめに

計算機の処理速度向上や安価なストレージ、家庭への通信インフラ整備によって、ADSL をはじめブロードバンド接続によるコンテンツ配信が行われつつある。しかし現状ではグローバルなネットワークにおいてファイルサイズの大きいマルチメディアコンテンツを配信するには十分な状況とはいえず、配信効率 (ネットワーク負荷軽減・大容量ファイルの転送遅延改善・ストリーミング安定配信など) を高めるために最適化されたネットワークとして、コンテンツ・デリバリ・ネットワーク（以降、CDN と表記）が構築、運用されている。CDN ではコンテンツを効率よく配信するためにキャッシュ、ルーティング、コンテンツ管理手法などを最適化する。CDN インフラを利用したコンテンツ配信サービスの例として、Akamai Technologies [14] 社はキャッシュ技術などを最適化した高速ネットワークで繋がるサーバ群を有して成功を収めており、今後こうしたコンテンツ配信モデルの需要はさらに高まっていくと考えられる。
一方、ピア・ツー・ピア (Peer to Peer、以降 P2P と表記) 環境では Napster [18] に始まった Gnutella [17] や Winny [19]、Freenet [8] などのコンテンツ共有アプリケーションが個人ユーザ間で利用されている。これらのサービスは TTL(Time to Live) でクエリー制御することによって、正常機能するネットワーク上に配置されるコンテンツすら確実に発見できると保障せず、さらにインデックス管理の面で帯域を消費するなどによってスケーラビリティを確保できない。これに対して Pure な P2P 環境で利用可能な分散ハッシュテーブル (DHT:Distributed Hash Table) [1]、 [2]、 [3]、 [4]、 [7]、 [12](第 1.2 節を参照) を利用する方法が注目されている。
筆者は、このような P2P コンテンツ管理手法を最適化することで、CDN について近年行われているサーバネットワーク構築以外の手段を提供できると考えている。
そこで本研究では、PureP2P 環境を対象とした、戦略的なコンテンツ管理を可能とするルーティングアルゴリズムを提案する。具体的には、実ネットワーク上の地理

情報やノード間の距離情報を収集し、それをノードの ID に反映させた DHT を提案、実装する。本稿では、提案手法の概念や実装方法について述べた後で、実験・考察を述べ、結論を述べる。

1.2 研究背景

DHT は、P2P 環境で位置非依存な GUID を用いて構築したオーバレイネットワーク上でクエリールーティングすることによって位置透過的なコンテンツ発見 (分散コンテンツ発見) を可能とするもの (第 3 章参照) で、ユーザは欲しいコンテンツの GUID を入力することで正常機能するネットワーク上に存在するコンテンツを得ることができる (図 1.1 参照)。ここで、コンテンツ発見のオーバレイネットワークでのホップ数は一般的に最大 $O(\log N)$ (N:ノード数) だと知られている [10]。またインデックスを分散配置する機構 (第 3 章参照) によって、ノードの動的参加・脱退への柔軟な対応 (第 3.3 節参照) や、ネットワーク負荷集中の回避 (第 3.5 節参照) を実現する。Tapestry [1] や Pastry [2] などはスパニングツリーを応用してクエリー制御することでこれを実現している。

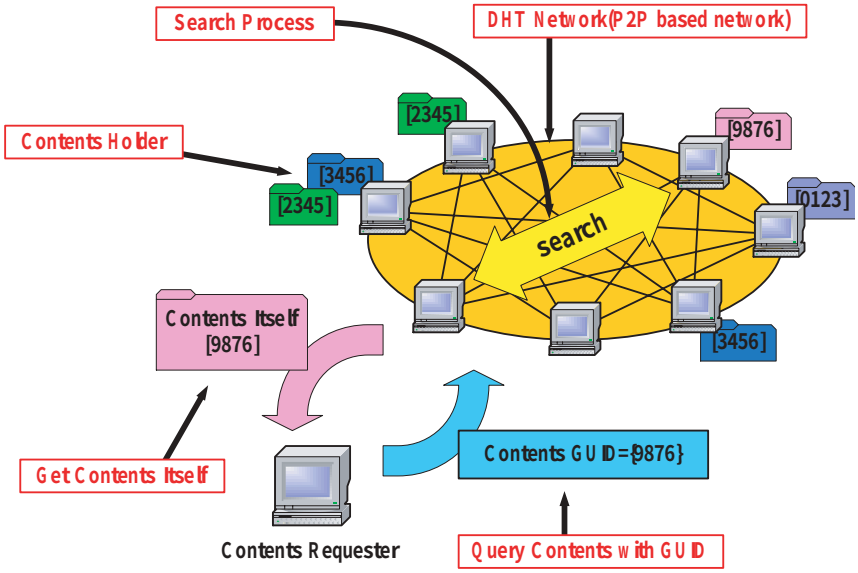

図 1.1: DHT における GUID 入力によるコンテンツ取得

1.3 研究目的

Tapestry をコンテンツ配信に用いると戦略的なサービスを実現できるが、コンテンツ管理について様々な問題が挙げられる。具体的には以下の項目について施策を立てる必要があると考える。

α 特定コンテンツへのホットスポット回避

β 災害等で一斉にノードがネットワーク脱退によるコンテンツ消滅回避

γ ユーザのアクセス効率向上

Tapestry のような DHT を利用することでネットワーク上に存在するコンテンツを確実に発見できるようになるが、α のように、特定コンテンツへのアクセス集中が起こる可能性がある。Tapestry ではこれについて施策を実装しているが (第 2.2 節参照)、これはホットスポット発生を検知する手法であり、ホットスポット発生前に戦略的に配置する施策を用いるべきだと考える。β については、例えばコンテンツ配信プロバイダが DHT 上でシステム運用することを考えれば、コンテンツのアベイラビリティを高めるために必要な施策であり、γ についても β と同例を考えると、ユーザへのコンテンツ配信遅延の削減、ネットワーク負荷軽減等に必要な施策だと考える。そこで本研究の最終目標は、これらのコンテンツ管理に関わる問題を解決する戦略を考える上で、戦略に用いる有用な情報を提供できるオーバレイネットワークを構築することである。評価については、オーバレイネットワークが有効な情報を提供する精度を測る。

1.4 研究の実装と評価

本研究では、前節で挙げた問題の解決に 2 つの指針を掲げる。

α 実ネットワークの地理情報やノード間の距離情報を考慮した ID 付けを行ったノードで DHT を構成する

β Tapestry で実現されているルートノード (第 3.5 節参照) という機構を応用して情報収集

Tapestry では、隣接ノード同士の関係性として、ランダムに生成されるハッシュ値のプレフィックス一致桁数を用いていたが、このハッシュ値にノードの関係性を埋め込むことで、隣接ノード間に明確な関係性を定義したオーバレイネットワークを構築することができる。α は具体的には、ノードの存在位置を一定区域ごとに分けて

ID 付けすることを目指す。これによって、例えば全ネットワークで均等にコンテンツを配置する、などということが可能になる。β を実装することで、ルートノードが DHT ネットワークの状況を把握でき、戦略的な行動を取ることが可能になる。上記 2 点を実装したシステム、Zone Overlay Network を構築し、これを利用した分散コンテンツ発見アルゴリズム Zone-DHT を実装する。
このように Zone-DHT は実ネットワーク位置情報を反映したオーバレイネットワーク上で情報収集することによって戦略的な手法を実行できる点が従来 DHT に対する優位性として挙げられる。
Zone-DHT の評価としては以下の実験を行う。

- 隣接ノードの関係性をどれだけ示せているかを測定する
- 各ノードがオーバレイ上のノードの隣接情報を収集する精度を測定する

1.5 研究の想定サービス

本研究が想定する環境について以下に列挙する。

- 映像などのマルチメディアコンテンツをインターネットを介して配信するサービスを想定
- システムストレージとしてサービス提供を望むユーザの固定端末を使用し、ユーザはストレージの一部をネットワークに公開する
- DHT ネットワークを使用

1.6 論文の構成

本稿は以下の 7 章からなる。

第 1 章　序章
本論文の概要、目的、構成について述べる。

第 2 章　関連研究
従来の関連研究を紹介し、本研究との相違点を述べる。

第 3 章　Tapestry オーバレイの解説
提案手法の説明に先がけ Tapestry オーバレイの機構を簡単に説明する。

第 4 章　地理属性を考慮した Zone-DHT とその応用

提案手法の実現方法について述べる。

第 5 章　実装

提案手法の実装について述べる。

第 6 章　評価と考察

提案手法の評価を行い、その結果を考察する。

第 7 章　結論

本稿の結論と、本研究の今後の課題を述べる。

第2章　関連研究

本章では、コンテンツ配置に関する従来研究を幾つか紹介し、それらと本研究の相違点について述べる。

2.1　Yan Chen等の研究、Coral、Akamai

Yan Chen 等 [5] は、Web サイトのコンテンツレプリケーションについてプル型 (Un-cooperative Pull、クエリー状況から必要に応じて作成されるコンテンツ配置) とプッシュ型 (Cooperative Push、ユーザのアクセスパターンなどから事前にコンテンツ配置) に分類し、この優劣を比較しており、プッシュ型の方が戦略的な配置が可能で、通信コストや管理コストが制御できることや、新規投入コンテンツについてアベイラビリティなどを改善できる、と示している。また、「分散した全 Web コンテンツレプリカから全てのクライアントへの距離のうち最も近距離の総和を最小にする」という目標が数式で表されており、本研究の参考にできる。しかしここでは、Web コンテンツ全般を対象としており本研究のマルチメディアコンテンツを想定するとサーバのストレージ容量や通信料などがボトルネックになる。またサーバネットワークを想定しており、本研究の PureP2P 環境と異なる。Coral [7] や Akamai [14] も同様である。

2.2　Tapestry ホットスポット回避

Tapestry [4] は、環境変化を検知して自立的に自己組織を最適化することを 1 つの目的としており、その意味でホットスポットが発生した際の施策、*hotspot cache* を提案している。具体的には特定コンテンツに短期間に多数クエリーされると、それらは最終的にクエリールーティングの到達地であるルートノード (3.5 節参照) とコンテンツホルダに集中するので、これに対してコンテンツが常に自身へのクエリー数を監視し、集中が起こった場合にルートノードに報告することで、ルートノードが適当なノードに対してレプリカを生成する。これは Yan Chen 等の研究 [5] でいうプル型レプリケーションであり、あらかじめ戦略的なプッシュ型コンテンツ配置を実現

させる本研究とは異なる。

2.3 中内らの研究

中内らの研究 [9] では、関連コンテンツをスケーラブルかつ高速に発見するためにオーバレイネットワーク上で関連コンテンツを近隣に配置するアルゴリズム、$CDHT$、を提案している。本研究とテーマは異なるが、ここで用いられる階層型 GUID の思想は本研究に参考になる。

2.4 Jeffrey の研究

Jeffrey [13] は Chord [12] オーバレイネットワークトポロジを最適化する手法としてクラスタリングを提案しているが、このような最適化が本研究で提案する Zone ルートノードの機能拡張として提案できる可能性がある。

第3章　Tapestryオーバレイの解説

本研究ではTapestryオーバレイを応用してシステムを実装する。そこでまず本章では、従来DHTとしてTapestryのアルゴリズムを解説する。

3.1　概要

DHTでは、全てのコンテンツとノードにそれぞれユニークなGUID（Globally Unique Identifiers）、NodeIDが割り当てられる。GUIDにはコンテンツ名のハッシュ値などが用いられる。NodeIDにはIPアドレスやMACアドレスのハッシュ値が割り当てられる。ハッシュ関数には、SHA-1などのハッシュ関数を用いる。こうして定義されたノードを用いてNodeIDに基づいてP2Pオーバレイネットワーク[1]が構築される。このネットワークにおいて、各ノードはクライアントもしくはコンテンツをストレージに保持したコンテンツサーバ(以降、コンテンツホルダと表記)、もしくはメッセージ中継ハブとして動作する。コンテンツホルダ情報は、コンテンツのGUIDと一致するかもしくは一番近いNodeIDを持つノードによって管理され(第3.5節参照)、ユーザがコンテンツをクエリーする際には、このノードにメッセージを送信することによってコンテンツを発見する。ここで位置非依存なNodeID、GUIDを用いてクエリールーティングを行うことにより、位置透過的、複製透過的コンテンツ発見が可能となる。コンテンツ発見に必要なオーバレイネットワーク上のホップ数は一般的に最大$O(\log_\beta N)$ (β:NodeIDの一桁がとり得る値の数、N:ノード数)だと知られている。

3.2　リンク構造

各ノードは、通信先ノードのNodeIDに数値が近くなっていくようにメッセージ処理するルーティングテーブルを持つ。テーブルには自身のNodeIDと上位桁の一

[1]オーバレイネットワークとは、各サービスやアプリケーションの目的に応じた繋がり方を実現するネットワークのことを指し、ここではIPネットワーク上に構築される論理ネットワークのことである。

致 (0≤i≤([NodeID 桁数]-1)) によってレベル ($L[i+1]$) 分けされた論理リンク情報を持つ。図 3.1 は GUID、NodeID 共に 16 進数 4 桁で表す場合のリンク構造の例を示している。各レベルには、一桁の数値がとり得る値 (この場合だと 16 通り) の数のノードを格納できる。

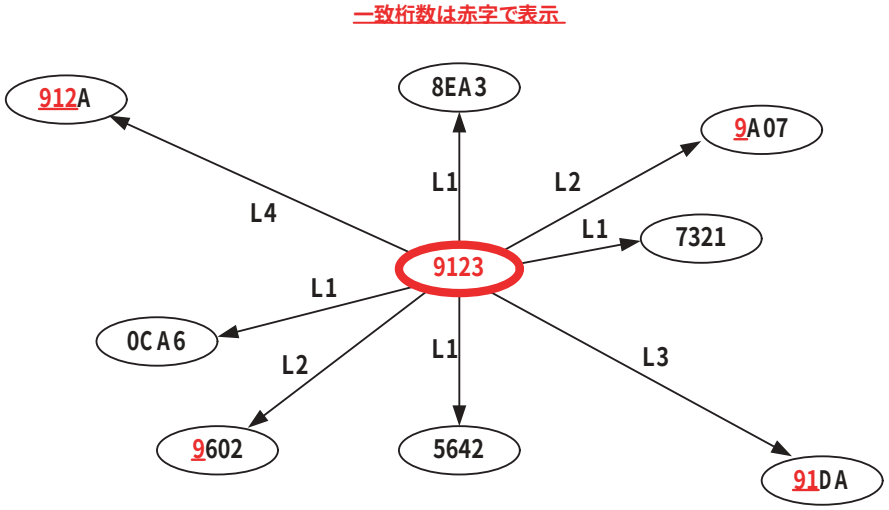

図 3.1: リンク構造例 (NodeID[9123])

3.3 ルーティング

本研究のルーティングアルゴリズムがベースとする、Tapestry のルーティングアルゴリズムを示す。
Tapestry 上でのルーティングは、一言で言うと、ID 空間の中でそのノードの NodeID に次第に近くなっていくようなリンクを通過する、と言える。具体的には、経路上の各ノードにおいて、メッセージ送信先ノードの NodeID と一致するプレフィックス桁数が 1 ずつ増加するようなノードが、メッセージ転送先としてテーブルから選択される。このメッセージ経路を、NodeID[CA12] から NodeID[5027] へのメッセージ送信を例にして説明すると、メッセージは

$$NodeID[CA12] \rightarrow NodeID[5DA5] \rightarrow NodeID[505C] \rightarrow NodeID[5021] \rightarrow NodeID[5027]$$

という経路を辿る (図 3.2 参照)。このようなルーティング手法を本稿では Tapestry ルーティングと呼ぶことにする。Tapestry オーバレイの特長として、ノードの動的参加・脱退への柔軟な対応が挙げられるが、障害などによってノードが突然脱退さ

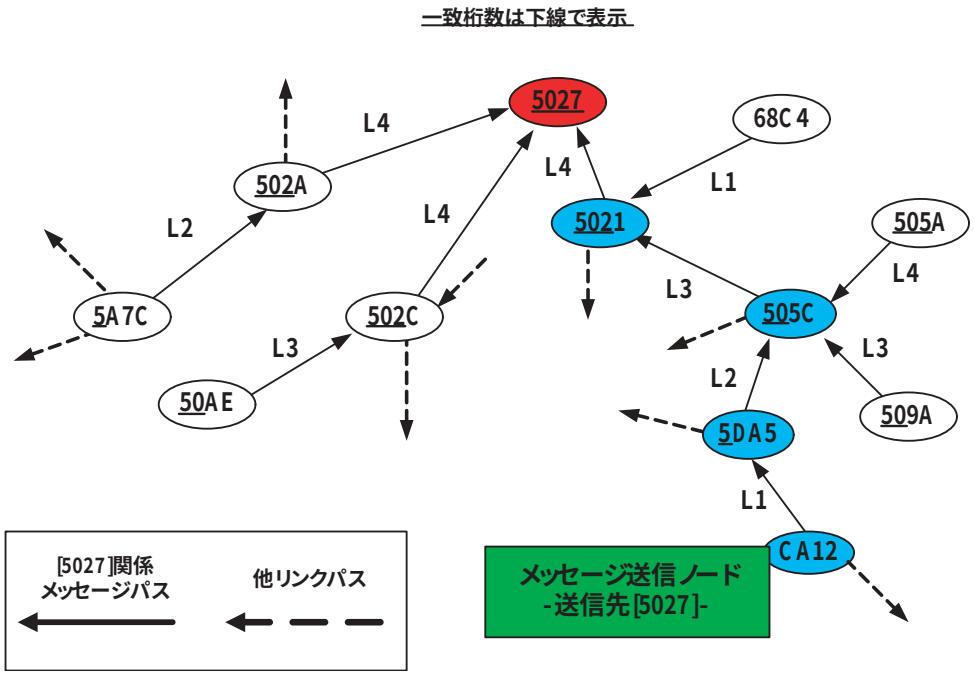

図 3.2: Tapestry のルーティング例

れた場合にも目的のコンテンツが発見できることを図 3.3 に示す。先ほどのネットワークにおいて NodeID[5DA5] が障害等によって突然ネットワークから脱退した場合でも、目的ノードに到達できる例を示している。このように、NodeID[CA12] が NodeID[5DA5] の脱退を検知しても、代替ノードにメッセージを送信することで目的ノードに到達できる。

3.4 スタビライズ処理

Tapestry オーバレイでは、ノードの生存を定期的に確認するため、ルーティングテーブルに登録されているノードに対して定期的にビーコンを送信する (以降、スタビライズ処理と表記)。

3.5 ルートノードによるコンテンツ管理

コンテンツホルダ情報はコンテンツの GUID と一致するかもしくは一番近い NodeID を持つノードによって管理される。これをルートノードと呼ぶ。ルートノードを利

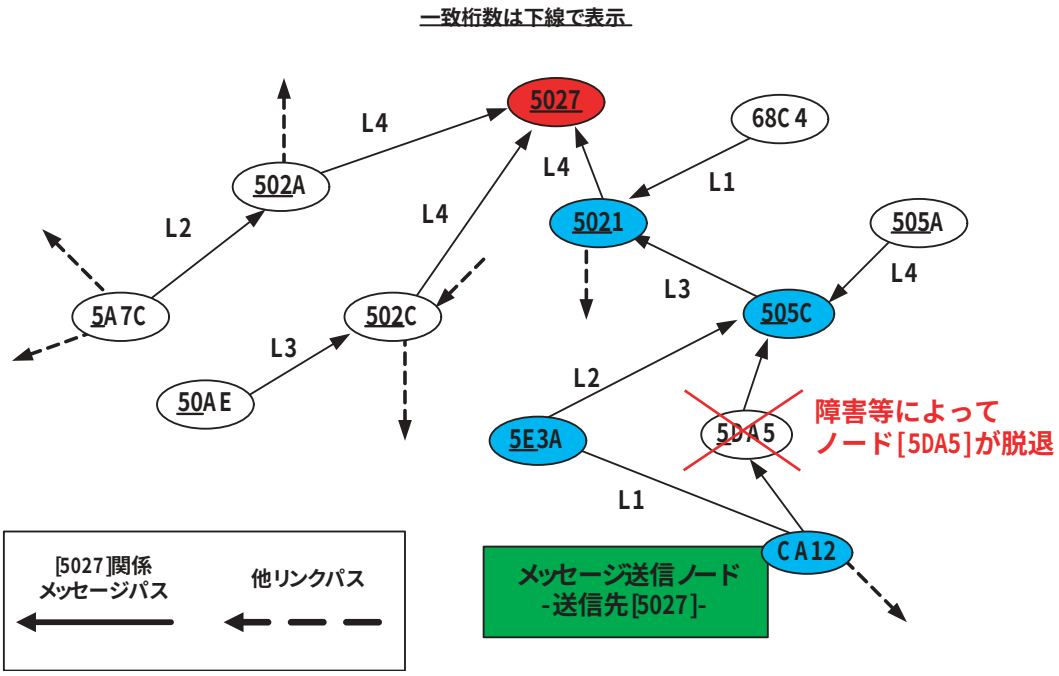

図 3.3: 突然のノード脱退時のルーティング例

用したコンテンツ管理手法について説明する。

3.5.1 クエリールーティング

クエリールーティングはルートノードに Tapestry ルーティングによってクエリー送信することで行われる。ルーティングは、ルートノードに到着するかその手前で目的のコンテンツを発見するまで継続される (図 3.4 参照)。ここで、ルートノードまで到着しても目的のコンテンツを発見できない場合、そのコンテンツはネットワークに存在しないということが分かる。尚、このルートはスパニングツリーを形成するルーティングアルゴリズムが用いられている。

3.5.2 パブリッシュルーティング

クエリーをルートノードまで中継するノードはそれぞれ、自身でコンテンツホルダ情報 (以下、インデックスと表記) を収集しており、それは [コンテンツホルダ識別情報、$GUID$] の組で構成される。この情報は、コンテンツホルダがネットワークに自身の情報を定期的に登録する、パブリッシュ動作の際に収集される。パブリッシュ

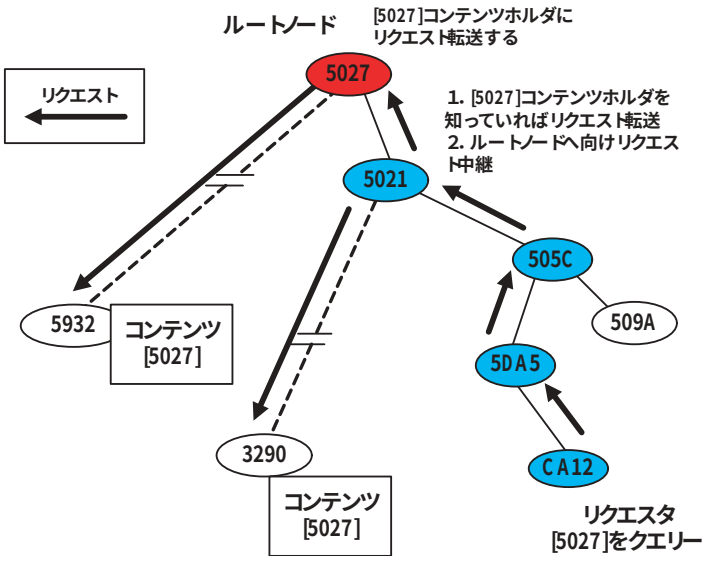

図 3.4: Tapestry のクエリールーティング

動作ではメッセージを Tapestry ルーティングによってコンテンツのルートノードに向かって送信する。この経路は、コンテンツホルダがクエリーを送信した場合と同経路を辿る。このときパブリッシュメッセージを中継するノードは、自身にインデックスを登録する（図 3.5 参照）。このようにインデックスはネットワークに分散配置される。以降本稿では、クエリーの際にリクエスタからルートノードに到着するまでの経路でメッセージを中継するノードをキャッシュノードと呼ぶことにする。

3.6 Tapestry における各ノードの役割

Tapestry においてのノードの役割としては、ルートノード、キャッシュノード、コンテンツホルダ、リクエスタが挙げられるが、実際にネットワークを構成する各ノードの能力は等しく、ノードは NodeID によって各コンテンツに対して様々な役割を担うことになる。

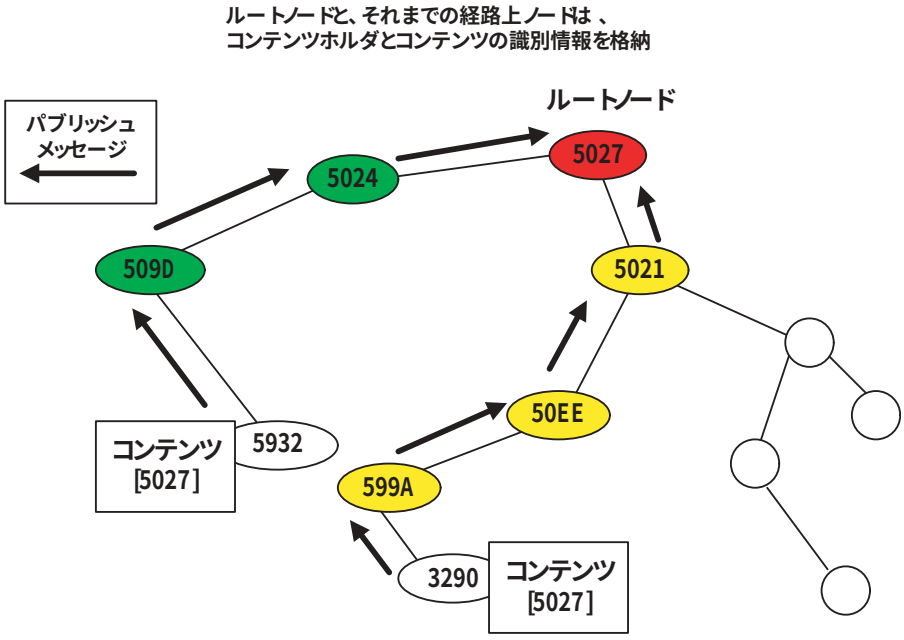

図 3.5: Tapestry のパブリッシュルーティング

第4章　地理属性を考慮した Zone-DHTと 戦略的コンテンツ配置への応用

本研究では、各ノードについて近隣ノードとの距離情報を考慮したID付けを行うオーバレイネットワーク、Zone Overlay Network(Zone-DHT) を提案、構築し、オーバレイ上でノード情報を収集して戦略的コンテンツ配置 (Zone Overlay Replication) を実現する。そこで本章では、Zone Overlay Network及びZone Overlay Replicationアルゴリズムの内容を述べる。

4.1　Zone Overlay Network

本節では、Zone Overlay Networkアルゴリズムについて述べる。

4.1.1　概要

Zone Overlay Networkとは、従来のTapestryオーバレイの応用として、物理ネットワークの位置情報やノード間の距離情報を考慮したID付け行ったオーバレイのことである。具体的には、各ノードにあらかじめZONE判定距離情報を持たせ、通信の際に取得したノード間の距離がその区分に収まる場合に、それらの存在地域が同一であることを示すID付けを行い、ネットワークを脱退・再参加させる機構を持つ。これによって、実ネットワーク的に近距離に位置するノードのクラスタ (Zone Cluster、以降ZONEと表記) がオーバレイ上に生成される。各ノードはこうした処理を繰返すことで最終的に自身の所属すべきZONEへ参加する。このような位置情報を考慮したオーバレイとTapestryオーバレイとの構造的な違いを図4.1を用いて解説する。図4.1において真中のネットワーク図は、オーバレイ参加ノードの物理ネットワークで、仮に三拠点のノードがオーバレイに参加することを示し、上下のネットワーク図は、ノードがオーバレイに参加した際の論理ネットワーク構成例を示している。上図

は、Tapestry オーバレイに登録されることで物理ネットワークとは無関係にノードがリンクされることを示している。このようなネットワークでは、隣接ノード同士の関係性が定義ができない。しかし下図のような、完全にノード地理情報を把握した Zone オーバレイでは、論理ネットワーク位置を参照しても自身に一番近いノードを発見できるような構造になる。これによって例えばある ZONE でホットスポットを発見した場合に同 ZONE や同傾向が予測できる別 ZONE に複製配置する、などの戦略的コンテンツ管理が可能になる。尚、Zone オーバレイでは、自身の近くのノードを発見できるが、Tapestry オーバレイと同様に他にも様々なネットワーク上のノードが登録される。

4.1.2 NodeID

各ノードは以下のような ZONE 情報を反映した階層型 NodeID を持つ。

$$NodeID = \{ZONE\_ID, UID\}$$

ここで ZONE_ID は ZONE を一意に識別するもので、ZONE 生成時のノードのうちいずれかの UID を使用する。UID は全オーバレイ上でノードを一意に識別するもので、IP アドレスや MAC アドレスなどを使用する。
どの ZONE にもあらかじめ登録されないノードがオーバレイに参加する場合は、DEFAULT_ZONE 所属ノードとして NodeID を登録される。DEFAULT_ZONE については後述する第 6 章において、DEFAULT_ZONE を一つだけ定義した場合と複数定義した場合で、NodoeID に偏りが発生するか、比較している。
Zone-DHT では、ZONE 生成 (第 4.2.2 節参照) の際に新 ZONE に所属するどちらかのノードの UID の ZONE_ID として登録する。このとき UID を全オーバレイ上で一意な値にすることで、ZONE_ID も必ず一意となる。

4.1.3 ZONE ルートノード

前節のような ID 付けを行うことで、Tapestry におけるコンテンツルートノードの仕組みを利用して、各 ZONE にルートノード (以降、ZONE ルートノードと表記) を定義できる。ZONE ルートノードの NodeID は以下のように定義する。

$$ZONE \text{ルートノードの } NodeID = \{ZONE\_ID, ZONE\_ID\}$$

ZONE ルートノードは、Tapestry のルーティング機構を活用するので固定ノードとして用意する必要はない。つまり Tapestry ルーティングのように、ネットワークの状

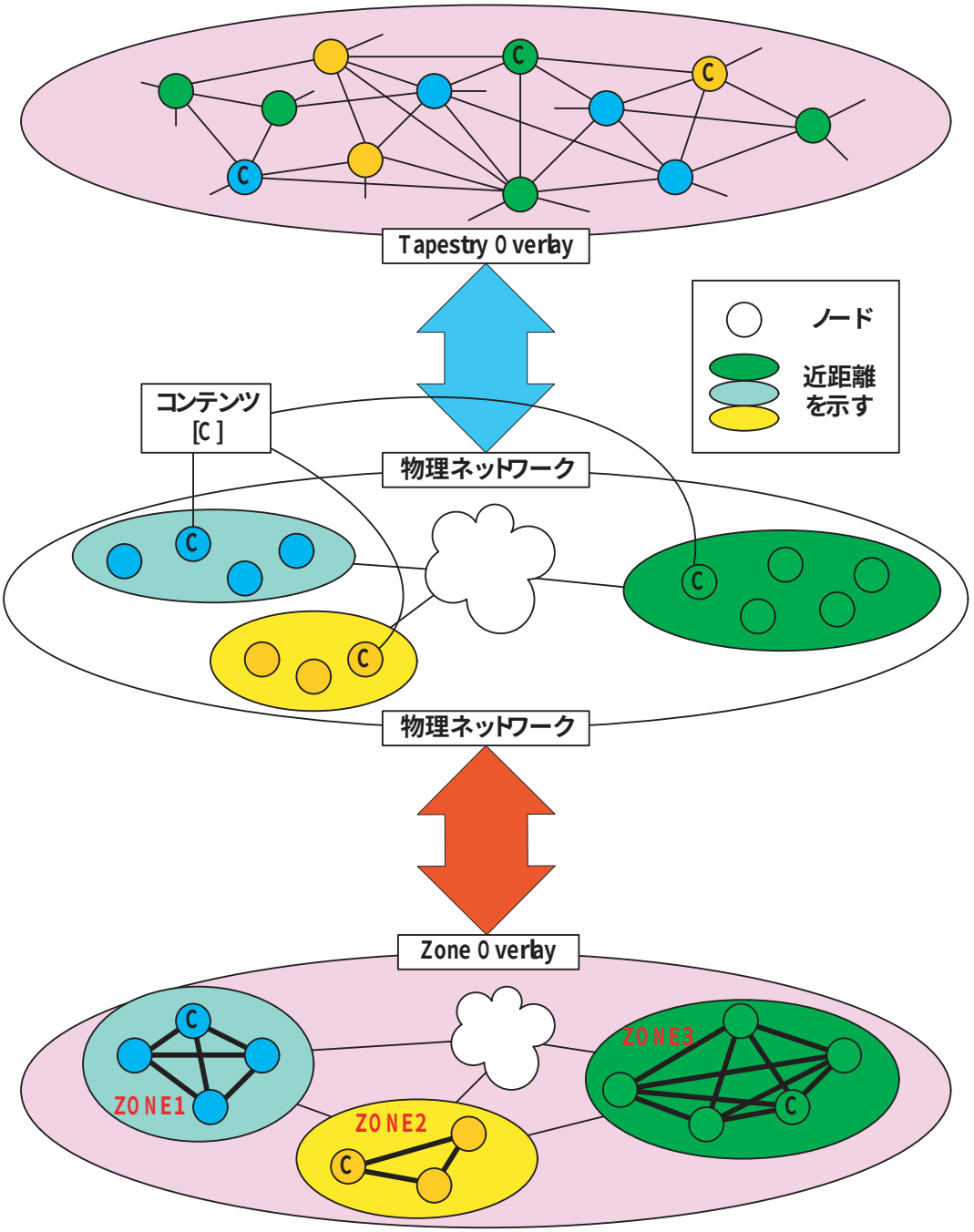

図 4.1: Tapestry オーバレイと Zone オーバレイのネットワーク構造比較

況変化に応じてZONEルートノードも動的に変更される、という機構を用いる。例えばZONE生成時のZONEルートノードは、構成ノードのうち完全一致するNodeIDを持ついずれかのノードによって存在する。その後このノードがネットワークから脱退した場合でも、ZONEルートノードのNodeIDに近いNodeIDを持つノードがZONEルートノードとなる。このZONEルートノードは、ZONEに関する情報取得やZONE参加時のメッセージ送信先として活用される (第4.3参照)。

4.1.4 ZONE_IDマスクルーティング

Zoneオーバレイでは、Tapestryルーティングを実行するときにZONE_IDをマスクしてTapestryルーティングを開始することで自ZONE範囲内だけに絞ったルーティング処理を実行できる (図4.2参照)。ここでZONE内のノードを発見する場合、以下の最大ホップ数で目的ノードに到達できる。

$$\log_\beta ZN (ZN : ZONE \text{構成ノード数})$$

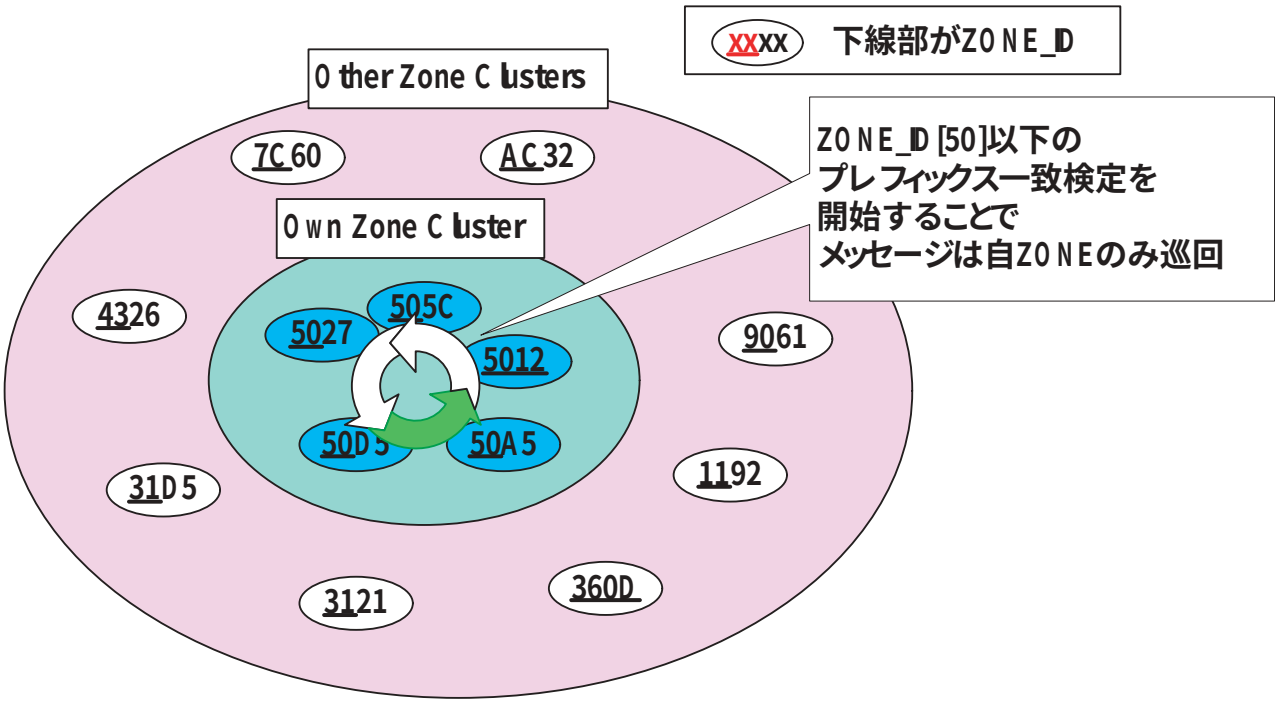

図 4.2: ZONE_IDマスクによるZONE内ルーティング

4.2 ZONE 管理手法

本節では、前節で述べた Zone Overlay Network における ZONE の管理手法について述べる。

4.2.1 ZONE 参加

各ノードは、自身の所属 ZONE(以下、自 ZONE と表記) を実ネットワークにおいてより正確なものへと変更するために、通信の際に通信先ノードとの距離 (ホップ数やパケットの RTT など) を計測する。このとき、通信ノード間の距離があらかじめ定義した ZONE 判定条件を満たした場合に ZONE 参加処理が実行される。処理手順は以下である。

1. 自身の ZONE_ID を参加 ZONE の ZONE_ID として、オーバレイを脱退・再参加する (以降、このように NodeID を変更してオーバレイに再参加する処理をノードリバイズと表記。ノードリバイズ処理の詳細については第 5.2.1 節参照)
2. ZONE_ID マスクルーティングによって ZONE ルートノードに参加メッセージを送信

2 番目について、ZONE ルートノード以外の ZONE 参加ノード (以降、ZONE 構成ノードと表記) に対しても参加通知を行える。

4.2.2 ZONE 生成

通信ノード間の距離があらかじめ定義した ZONE 判定条件を満たし、かつどちらも DEFAULT_ZONE に所属する場合に、ZONE は生成される。このときどちらかのノードが ZONE ルートノードに指名される。ZONE ルートノードが自身の UID を ZONE_ID としてノードリバイズすることで新 ZONE が生成される。ZONE 生成の通知を受けた相手ノードは ZONE への参加処理を実行する。

4.2.3 ZONE 併合

ZONE 生成処理はノード間で計測された距離情報を用いるので非同期に実行される。その過程では、本来 1 つの ZONE であるべきネットワークに複数の ZONE が生成される状況 (図 4.3 参照) が想定される。図の状況において、両 ZONE 構成のードのうち 1 組が、通信において互いの距離が ZONE 判定条件を満たすことを知ったとき、ZONE 併合処理が開始され、2 つの ZONE は合わさる。併合先としては、併合

処理に要するネットワーク負荷や時間を削減するために参加ノード数が多い ZONE に併合する。併合後 ZONE の ZONE_ID は併合先のものを使用する。
またここで、併合処理と同様に、距離情報が動的に変更されるものであれば ZONE の分離という手法も考えられる。

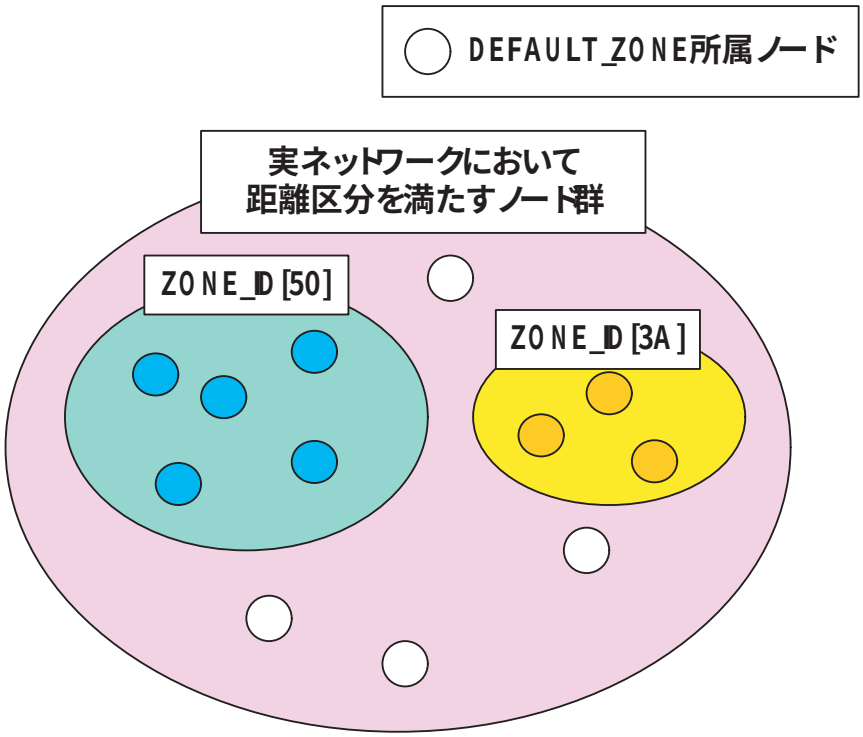

図 4.3: 実ネットワークで ZONE 判定条件を満たすノード群に複数 ZONE が存在

4.3 ZONE 情報の収集

本節では、ZONE 情報の収集手法について解説する。
ノードリバイズによって ZONE 生成、ZONE 消滅が起こるが、これらはオーバレイ上の全ノードが自身に一番適した ZONE に属するまで続けられる。そこで各ノードは最新の ZONE 情報を収集する。このとき ZONE 情報を、ノードの所属する ZONE(以下、自 ZONE) と他 ZONE に分類し、別の収集処理を行う。収集フェーズを分けることの優位性は、ZONE_ID マスクルーティングによって ZONE 情報収集に要するネットワーク負荷を軽減できることにある。

4.3.1 自・他 ZONE 情報の収集

自 ZONE、他 ZONE 情報の具体的な収集方法を、図 4.4 を例に説明する。図中において、大きい円はオーバレイ全体、小さい円はオーバレイ上の一つの ZONE、矢印はノード間の通信、矢印上のメッセージは通信に添付される ZONE 情報を示している。

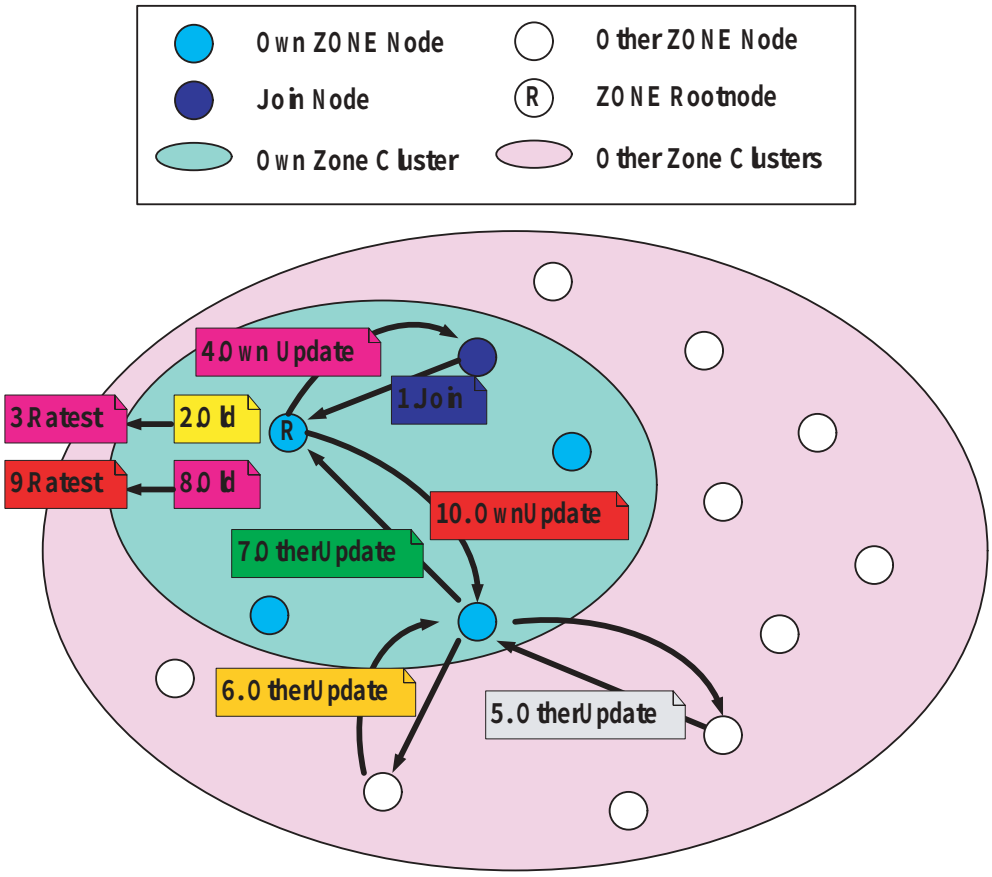

図 4.4: 自・他 ZONE 情報の収集

処理 1 ZONE にノードが参加する。参加メッセージが送信される。

処理 2,3 参加メッセージを受信した ZONE ルートノードは、自身のルーティングテーブルや ZONE 情報に参加ノード情報を加える。

処理 4 ZONE ルートノードは自身の持つルーティングテーブルと自 ZONE の情報を参加ノードへと返す。この時点で参加ノードについては ZONE ルートノードと参加ノードからの経路上に存在していた ZONE 構成ノードしか知らない。

所属 ZONE_ID[2F3D]
NODE_ID
B23A
3590
A412
・
・

表 4.1: 所属 ZONE の情報

処理 5,6 各 ZONE 構成ノードは、任意のタイミングでコンテンツ配信に関するルーティング等によって任意のノードと通信する。その際、他 ZONE 情報を通信相手から取得し、自身の情報に加える。

処理 7 ZONE 構成ノードは ZONE ルートノードに対して自 ZONE の更新情報を要求する (以降、ZONE スタビライズ処理と表記)。その際自身の持っている他 ZONE の情報やルーティングテーブルを添付する。ここで送信メッセージは、ZONE_ID マスクルーティングを用いることで、ZONE 構成ノードに対して自身の収集情報を提供する機会を与えることができる。ZONE スタビライズ処理は定期的に実行される。

処理 8,9 ZONE スタビライズメッセージを受信した中継ノードもしくは ZONE ルートノードはメッセージから相手ノードの情報を取得し、自身の ZONE 情報に存在しない情報を追加する。

処理 10 ZONE ルートノードは自身の持つ自 ZONE の情報を ZONE スタビライズ処理実行ノードへと返す。ZONE スタビライズ処理実行ノードは、この時点で ZONE 参加ノードの存在を知ることができる。

ZONE ルートノードは、ZONE スタビライズ処理が行われることで全 ZONE 構成ノードの情報を定期的に収集できる。この情報を別テーブルで持つ場合、それは例えば表 4.1 のようなデータ構造になる。ここで、ZONE をコンテンツ配置に応用するためにコンテンツホルダ情報やストリームに関する情報を載せても良い。

4.3.2 削除リスト

ZONE 情報は併合処理などによって、以前の ZONE 情報は無効になる。この状況への対処として各ノードは ZONE_ID の削除リストを持つ。各ノードが自身の持つ削除リストに登録されている ZONE_ID の ZONE 情報を無効情報とすることで、オー

バレイ全体から無効情報を削除できる。
削除リストへの ZONE_ID の追加処理は、自身の所属する ZONE が併合などによって削除される場合、ZONE 構成ノードが削除リストにその ZONE_ID を加えることで行われる。その後削除リストは、ノードが通信の際に自身の削除リストを添付することで、オーバレイ上に伝達される。削除リストを受信したノードは、リスト中の ZONE_ID を自身の有効な ZONE_ID リスト及びルーティングテーブルから削除し、最後にリスト中の全 ZONE_ID を自身の削除リストに加える。

4.3.3 ZONE 情報の不整合

ZONE スタビライズ時に ZONE ルートノードが障害などの理由でオーバレイから脱退している場合、ID が ZONE ルートノードに一番近いノードに到着する。そのノードは、自身が ZONE ルートノードでないにも関わらず ZONE スタビライズメッセージが自身以降に転送されない場合に ZONE ルートノードがダウンしていると分かる。
ZONE 中の任意のノードが何らかの理由でダウンした場合には、ZONE ルートノードにて定期的に ZONE 構成ノード情報をリフレッシュすることで対応できる。

4.4 Zone Overlay Network の性質

本節では、前節で述べた仕様のネットワークの持つ性質について述べる。

4.4.1 Tapestry ルーティング中の NodeID 更新

Zone オーバレイでは、各ノードは ZONE が変更される度にノードリバイズ処理によってオーバレイに再参加を繰返す。ここで各再参加処理によって、オーバレイ上に存在する全ノードのルーティングテーブルの該当エントリが有効でなくなる。ZONE 変更が頻繁に実行される状況では、Tapestry ルーティングが正しく行われない可能性がある。
そこで Zone オーバレイでは、古い NodeID を参照したメッセージが宛先ノードへ到着すると、到着ノードは自身の NodeID と比べて古い NodeID を参照していると判定し、送信先ノードへテーブルエントリのリフレッシュと Tapestry ルーティングのやり直しを求める。この動作によって Tapestry ルーティングは正しく実行できる。
Tapestry ルーティング中に NodeID を更新することによる長所は、正しいルーティングが実行され、かつルーティングテーブルエントリがリフレッシュされることであ

る。短所は最大ルーティングステップが一般的に知られる $O(\log_\beta N)$ (β:NodeID の一桁がとり得る値の数、N:ノード数) 以上の値を取ることである。Zone オーバレイでは ZONE 変更処理の負荷が収束していくことを考慮し、本手法を適用する。

4.4.2 削除による ZONE 情報量の変化

第 4.3.2 節で述べた削除リストについて、削除リストを添付した先のノードでの ZONE_ID リストの情報量変化を考えると、以下列挙した内容から、通信において交換する ZONE 情報には必ず削除された ZONE_ID と新 ZONE_ID 情報が含まれているので、各ノードが持つ ZONE 情報の情報量は保たれる (図 4.5 参照)。

- 削除された ZONE の構成ノードは全て同一の新 ZONE へと所属変更していること
- ノードは ZONE 変更の際に自身の所属を新 ZONE_ID に変更し、削除された ZONE_ID を削除リストに追加すること

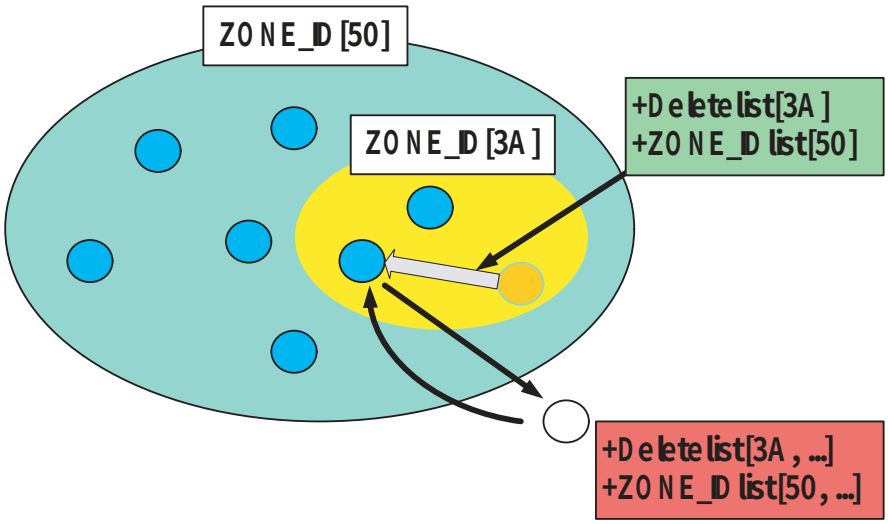

図 4.5: 削除リストへの追加と通信によるリスト更新

4.5 Zone Overlay Replication

前節のように管理される ZONE をコンテンツ配置に応用して、オーバレイ上に均等にコンテンツを配置するアルゴリズム、Zone Overlay Replication について述べる。

コンテンツホルダ		ZONE 情報
ZONE_ID	NODE_ID	ZONE_ID
2F3D	B23A	11C3
3AD8	3590	2F3D
5281	A412	5281
・	・	・

表 4.2: Zone Overlay Replication で用いるテーブル

4.5.1 概要

Tapestry オーバレイにおいて各コンテンツのルートノードは、コンテンツホルダのパブリッシュ処理 (第 3.5.2 節参照) の際にコンテンツホルダの情報を全オーバレイから収集可能である。コンテンツホルダによるパブリッシュルーティングや、ノードが障害でダウンするなど正常でない場合を除いて、ノードが所持コンテンツを削除する際に送信される削除通知によって、ルートノードは最新のコンテンツホルダ情報を知りえる。本手法ではこれに ZONE 情報を組み合わせる (表 4.2 参照) ことでコンテンツホルダの存在しない ZONE を発見し、複製配置を実行する。このとき、あらかじめ ZONE 間の距離を計測しておくことで、配置先 ZONE により近いコンテンツホルダから複製処理を実行できる。

4.5.2 コンテンツ配置のローカリティ

複製配置の際に、各 ZONE ルートノードに ZONE 構成ノード総数などの情報を問い合わせることで、各 ZONE 構成ノード数に応じた複製個数などを最適化することができる。
プル型コンテンツ配置として hotspot cache(第 2.2 節参照) と本手法を比較すると、前者では局所的に発生するホットスポットをその場で解決するだけなのに対して、後者では、局所的に発生したホットスポットについて他の地域でも同様の傾向が想定できる場合に、地理的に別の地域に複製をあらかじめ配置することが可能である。このように ZONE を利用したコンテンツ配置によって、各 ZONE の複製個数等を自律的に最適化することが可能になる。また本手法では戦略的なプッシュ型コンテンツ配置を実現することから、同数のコンテンツをランダム配置することに比べて、ホットスポット発生回避や局所的な災害時等の状況でのコンテンツ消滅回避、ユーザのアクセス効率向上の点で効果を挙げられると考える。

第5章　実装

本章では、前章で述べたZoneオーバレイの実装について解説する。具体的には、オーバレイ上の情報の定義、ZONE管理、情報収集について延べ、最後にZone Overlay Replicationについて述べる。尚、本章で挙げる処理フローでは、処理の大枠を示すためにTapestryルーティング中のNodeID更新処理は示さない。

5.1　Zoneオーバレイ上の情報定義

本節では、Zoneオーバレイに参加するノードのモジュール構成とZONE情報の構成について述べる。

5.1.1　ノードのモジュール構成

前章で述べたZone Overlay Network参加ノードは以下のモジュールで構成される。

α Tapestryルーティングテーブル

β 自身のZONE情報 (ZONE_IDとNodeID) とZONEスタビライズ機能

γ オーバレイ上のZONEリスト

δ コンテンツを保存するストレージ

ϵ ルートノードモニタリングと戦略的コンテンツ配置機能

ζ 自身の識別情報 (IPアドレスやMACアドレスなど)

η 全ての通信において自身の動的な識別情報を添付する機能

αはルーティングに必要である。βは自身のZONE情報を把握するものである。γはオーバレイ上に存在するZONEリストを管理するもので、これを利用してノードの地理情報を把握する。δは自身がコンテンツを保持するサーバとして機能するために必要である。ϵは、自身がコンテンツのルートノードであった場合に、そのコンテン

ツの配置戦略を行うためのものである。η について、ノードは自身の識別情報として静的情報と動的情報を持っている。前者は ζ にあたり、後者は所属する ZONE が変更されるという意味で、β における自身の ZONE 情報 (ZONE_ID と NodeID) である。オーバレイ上の通信によって適切な ZONE を発見するために、自身のそのときの ZONE 情報を添付している。

5.1.2 ZONE 情報の構成

各ノードが保持する ZONE 情報は、所属 ZONE と所属以外の ZONE 情報に分けられるが、これらは以下の要素から構成される。

α ZONE_ID をキーにした ZONE 情報リスト

β ZONE_ID の削除リスト

β については、他ノードとの情報交換によって自身が知ることのできた ZONE 情報だけを含む。ZONE 情報リストは以下の要素から構成される。

α 自身の ZONE_ID

β ZONE に所属するノードリスト

尚 β については、所属以外の ZONE については ZONE_ID だけを持ち、自身の所属 ZONE だけ持つ。つまり、所属 ZONE のノード情報を ZONE 内で共有する。これによって、自身が ZONE ルートノードである場合に全オーバレイ上からの ZONE 構成に関する要求に応答できる。
ZONE に所属するノードリストには、ZONE 構成を識別するための以下の情報が含まれる。

α ノード識別情報 (IP や MAC アドレスなど)

β タイムスタンプ管理による所持コンテンツリスト

γ タイムスタンプ管理によるサーバ・クライアント状態情報

β、γ についてはタイムスタンプで管理にすることで情報に一貫性を持たせてある。ZONE 情報を通信先ノードから受信したノードは、相手の ZONE 情報を自身に加える (以下、ZONE 情報マージと表記) 際に自身のノード情報を更新するかどうかの基準としてタイムスタンプ値が自身の値より最近のものであるかチェックできる。Zone Overlay Replication において β はコンテンツ配置先決定の際にそのノードを除外すべきか決定するために使用される。γ は β より優先度を低くするものの、同様に条件の 1 つとして使用される。

5.2 ZONE 管理手法

本節では、ZONE 管理に関する処理フローについて解説する。始めに ZONE 情報を変更する際に必ず行われるノードリバイズについて解説し、その後 ZONE 管理手法について解説する。

5.2.1 ノードリバイズ

まず始めにノードリバイズとは、ノードが一度オーバレイを脱退し、自身の NodeID を変更して再度オーバレイに参加する処理のことと定義する。本節では、この処理フロー (第 4.2.1 節参照) について解説する (図 5.1 参照)。

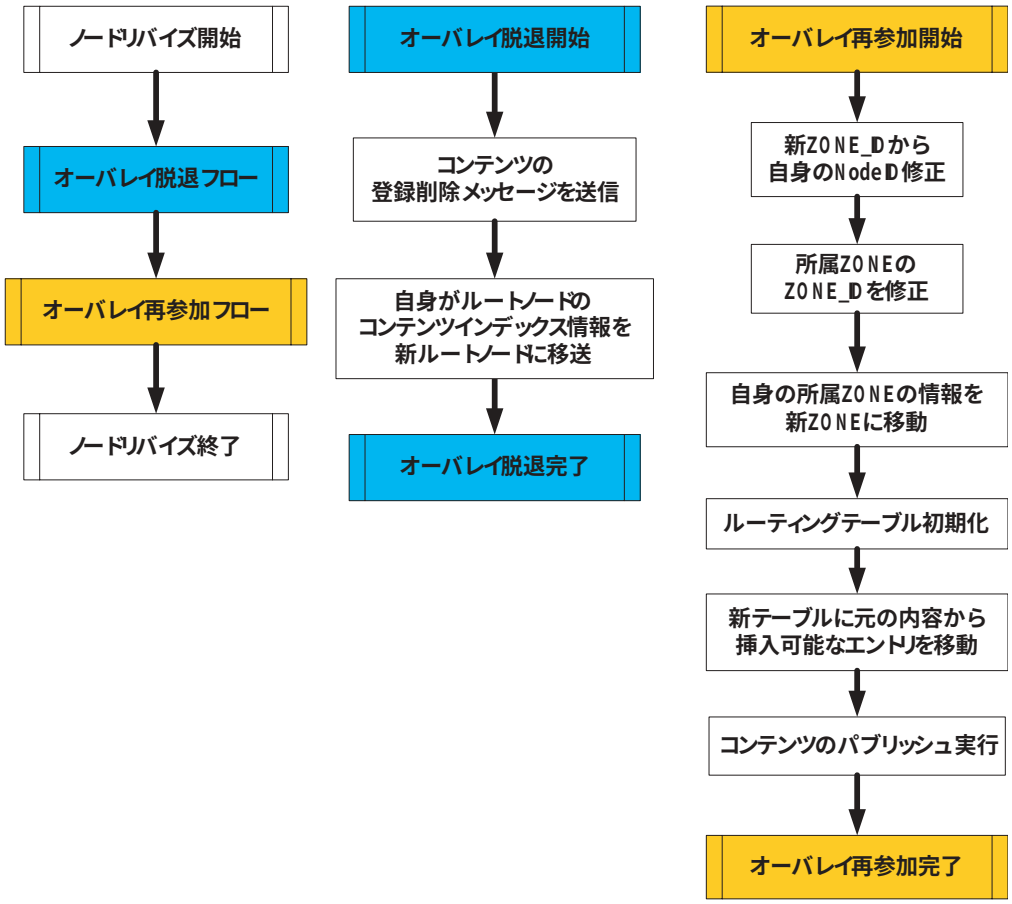

図 5.1: ノードリバイズフロー

ノードリバイズで重要な点は、オーバレイ上でノードを一意に識別する NodeID(=ZONE_ID,

UID) が変更されることである。オーバレイではノードの NodeID によってルーティング経路やコンテンツルートノードが設定されるので、同処理の際に変更前 NodeID がネットワークから消滅することによる影響を小さくするべきである。このような方針で NodeID を無効にする処理をオーバレイ脱退処理と呼び、新しい NodeID を有効にすることをオーバレイ再参加処理と呼ぶとき、ノードリバイズはこれら 2 つのフェーズに分けられる。
脱退処理では以下の二点に対処する。

1. 自身の保持するコンテンツについてのキャッシュ情報が無効情報になること
2. ルートノードの持つインデックスが無効になること

前者については、パブリッシュメッセージと同経路で登録削除メッセージを送信してインデックスをオーバレイ上から削除することで対応する。このときメッセージはコンテンツルートノードに対して Tapestry ルーティングによって送信される。後者については、自身がルートノードを努めている場合、そのインデックスを自身が NodeID を変更した後のルートノードに向けて送信する。この処理は、ルートノードがオーバレイ上で唯一、全コンテンツホルダのインデックスを管理している、という意味で重要である。
再参加処理では NodeID が変更されることによるルーティングテーブルの最初期化と自身の ZONE 情報の修正が行われる。ルーティングテーブルの再初期化では、以前の NodeID 時のテーブルエントリのほとんどが無効になる。よってこのとき、自身が ZONE を生成した場合でなければ ZONE 参加処理でテーブルエントリを増大させる処理を行う (第 5.2.2 節参照)。

5.2.2 ZONE 参加

処理フローでは、ノードリバイズした後で、ZONE_ID マスクルーティングによって ZONE ルートノードへの参加メッセージ送信が行われる (図 5.2 参照)。この処理フローを、参加ノードの視点でシークエンス図 5.3 で示した。

参加処理の目的は 2 つ、ルーティングテーブルのエントリを増加させることと、自 ZONE に関する情報を取得することであるが、Tapestry ルーティング中の全ての中継ノードからルーティングテーブルを、ZONE スタビライズ処理によって自 ZONE の情報を収集している ZONE ルートノードから自 ZONE に関する情報を得る。ここで ZONE 参加メッセージを受信した各ノードは、自身のルーティングテーブルと ZONE 情報に参加ノード情報を追加する。

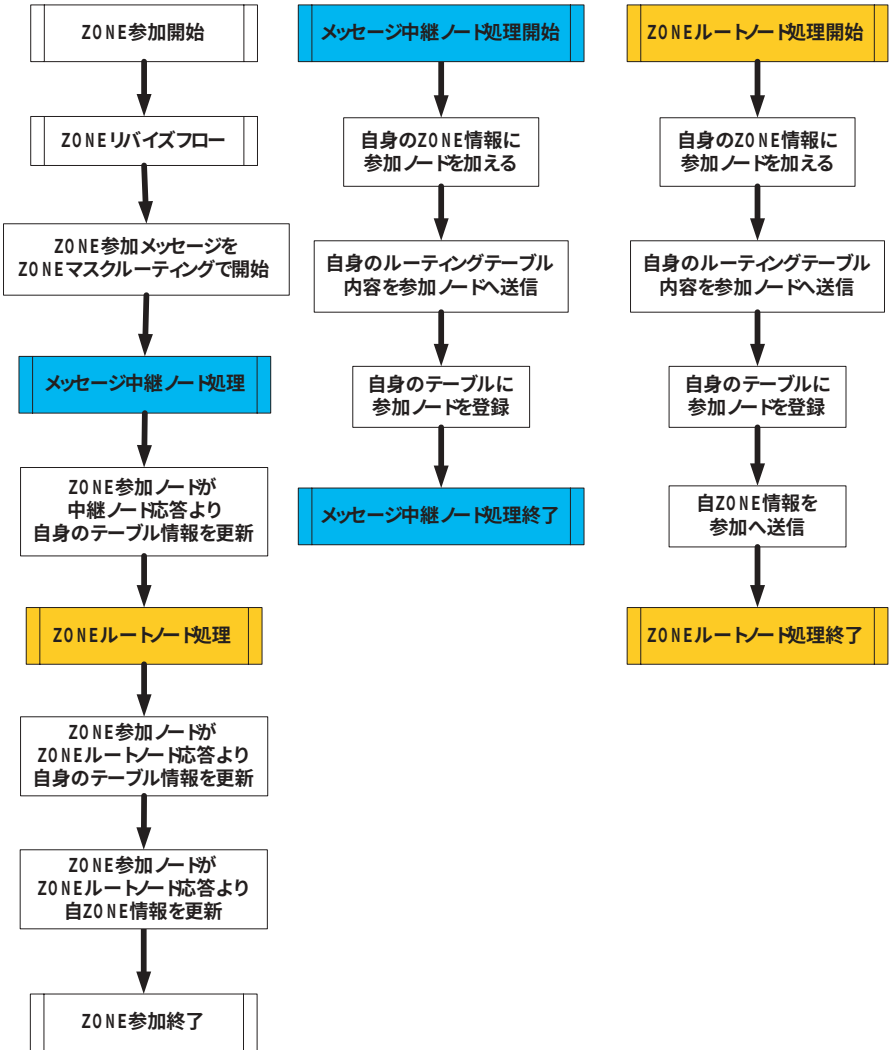

図 5.2: ZONE 参加フロー

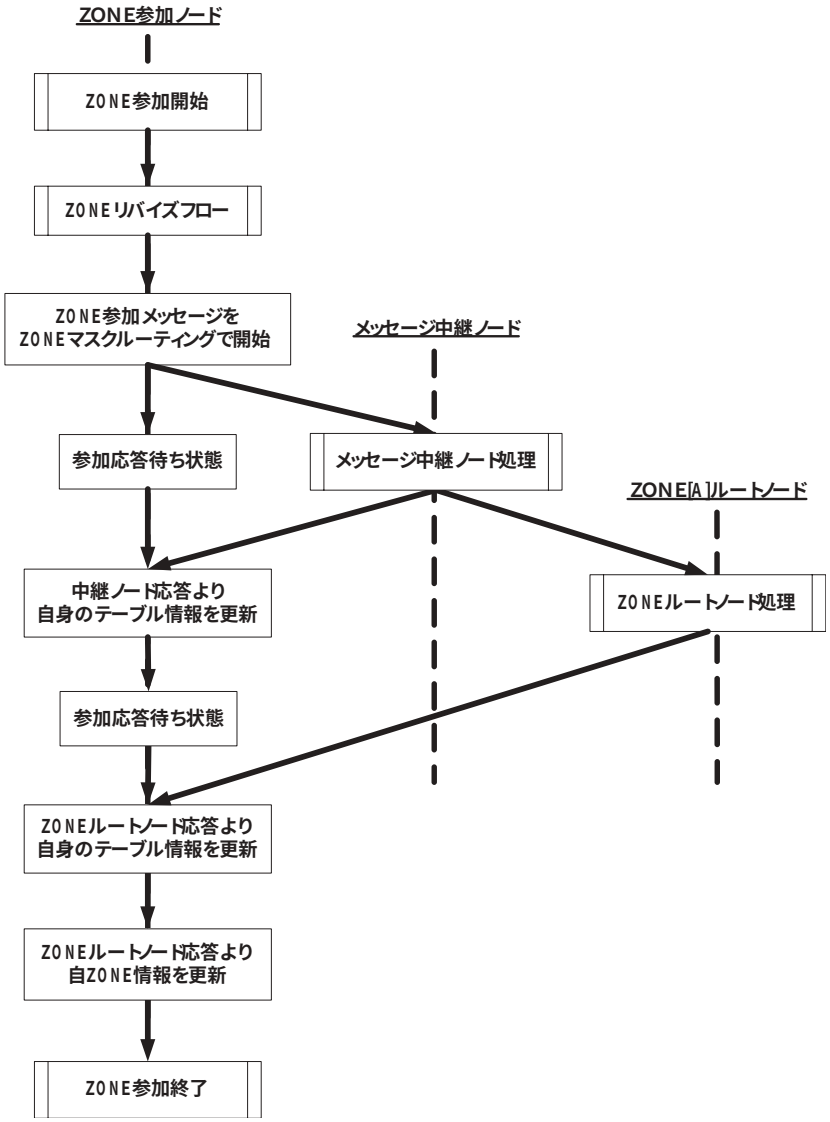

図 5.3: ZONE 参加シークエンス

5.2.3 ZONE 生成

通信ノード間の距離があらかじめ定義した ZONE 判定条件を満たし、かつどちらも DEFAULT_ZONE に所属する場合に、ZONE は生成される (図 5.4 参照)。そのとき ZONE ルートノードとなるノードは、ZONE 生成の条件が揃ったことに気付いたノードとする。このノードが ZONE_ID に自身の UID を指定してノードリバイズすることで ZONE は生成され、このノードはその後、ZONE 判定条件を満たした相手ノードに対して ZONE への参加要求を送信する。

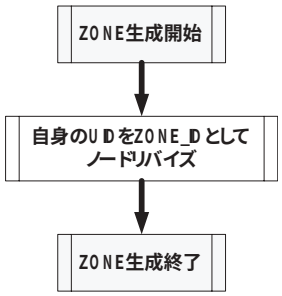

図 5.4: ZONE 生成フロー

5.2.4 ZONE 併合

併合フローを図 5.5 西メス。フローではまず、併合すべき状況を知った ZONE[A] 構成ノードが ZONE[A] ルートノードへ ZONE マスクルーティングによって併合通知メッセージを送信する。ZONE[A] ルートノードは、併合先 ZONE[B] の ZONE[B] ルートノードへ ZONE マスクルーティングによって ZONE[A] 参加ノード数を送信する。ZONE[B] ルートノードは ZONE[B] 参加ノード数を比べて、自身の方が多ければ ZONE[A] ルートノードに ZONE[B] 参加要求を送信、相手の方が多ければ ZONE[B] 構成ノードに ZONE[A] への参加要求を送信して、自身も ZONE[A] へ参加する。この処理フローをシークエンス図として表現したのが図 5.6 である。

以上の内容と前節で述べた ZONE 生成を合わせた ZONE 構成変更フローを図 5.7 に示す。

5.2.5 非同期の ZONE 構成変更

ZONE 参加及び併合処理は、関係ノード間のトランザクションである。これは、各ノードによって非同期に実行されるので、適切に管理されなければデッドロックなど

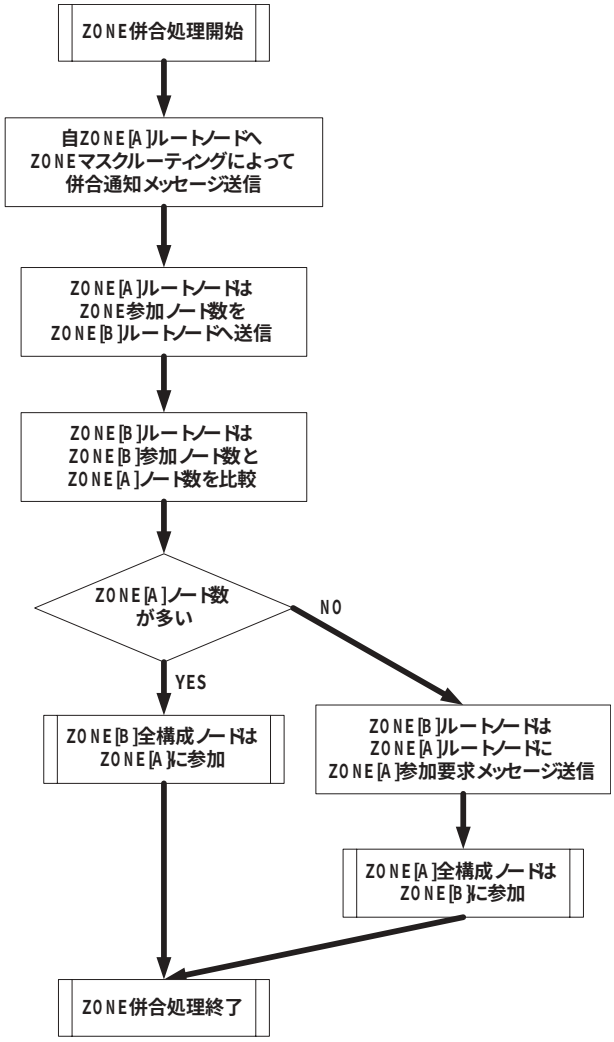

図 5.5: ZONE 併合フロー

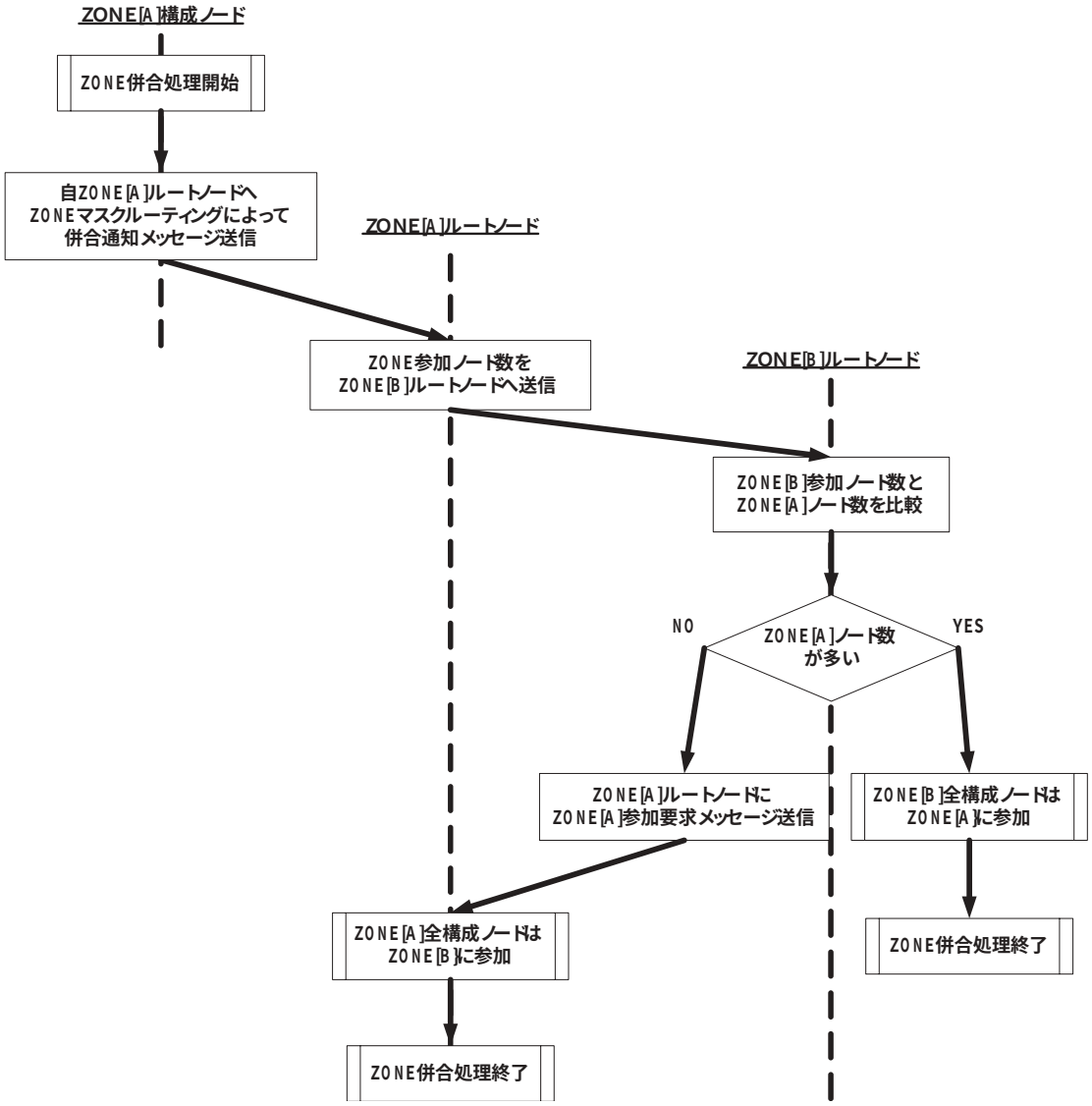

図 5.6: ZONE 併合シークエンス

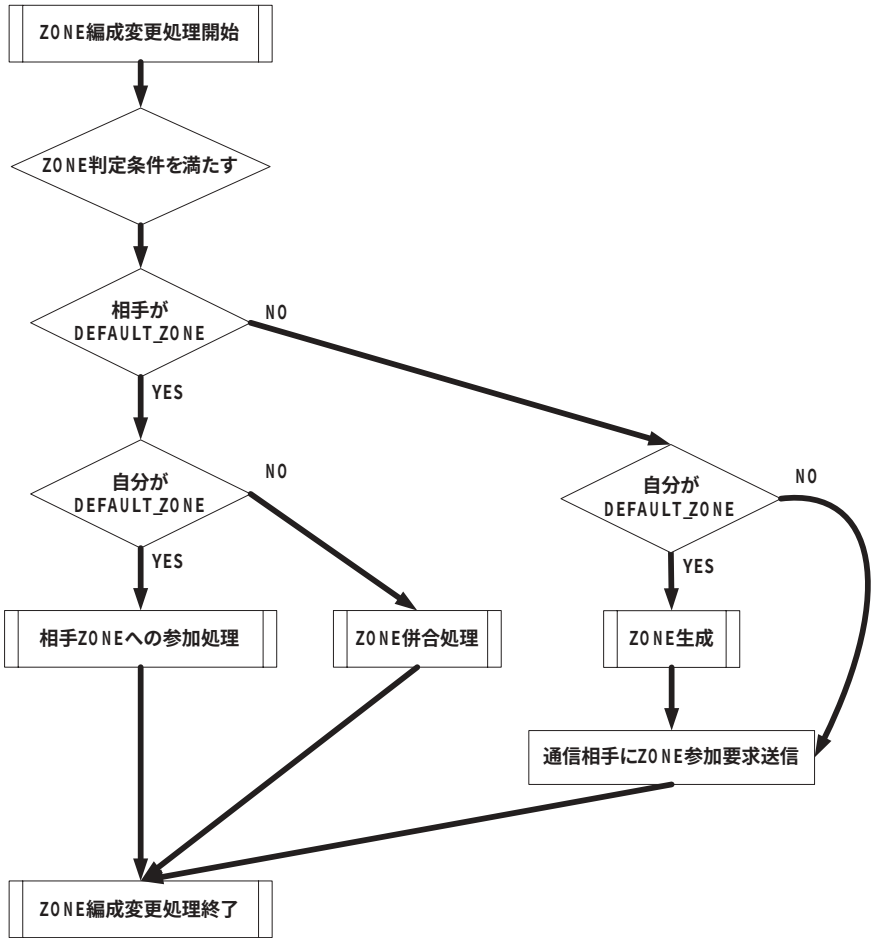

図 5.7: ZONE 構成変更フロー

の事態が予想される。本節では ZONE 参加及び併合処理について、トランザクション管理を実装した処理フローを解説する (フローは図 5.8、5.11 参照。シークエンスは図 5.8、5.9 参照)。

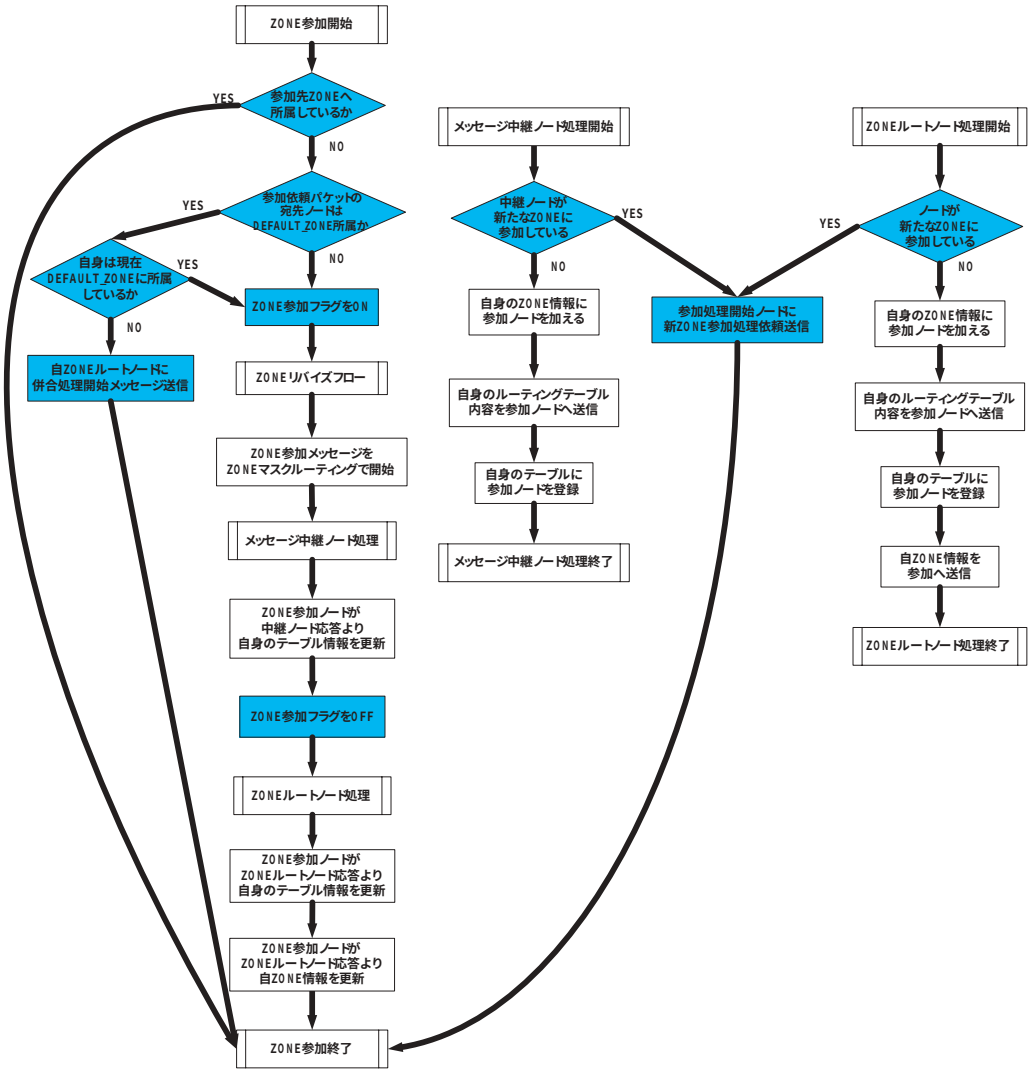

図 5.8: (改)ZONE 参加フロー

ZONE 参加フローの変更ポイントは 3 点、DEFAULT_ZONE に所属していたノードが参加する場合か、ZONE 参加処理中に他ノードが ZONE 参加依頼もしくは ZONE 併合処理を開始する場合に対処できるか、ZONE 参加メッセージ受信ノードの所属 ZONE が変更されているか、という点である。
DEFAULT_ZONE 所属ノードが参加する場合、参加ノードはまず自身の所属 ZONE

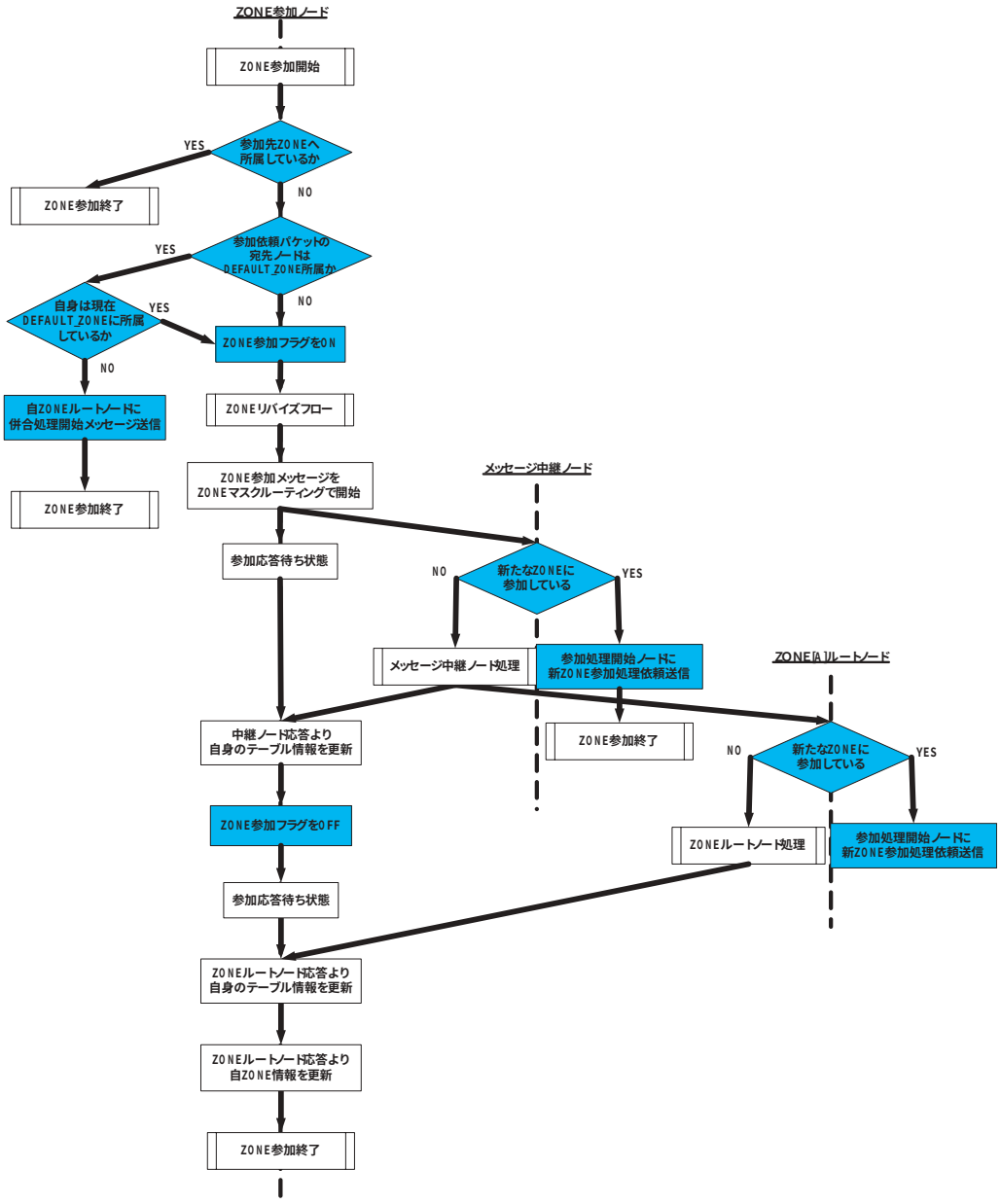

図 5.9: (改)ZONE 参加シークエンス

がDEFAULT_ZONEから変更されていないか調べる。変更されていた場合、それがこれから参加しようとするZONEならば参加処理を即終了し、そうでなければ自ZONEルートノードに併合処理依頼メッセージを送信して参加処理を終了する。
またここで、DEFAULT_ZONE以外のZONEに所属していたノードが参加する場合は、ZONEルートノードが併合交渉を行うため、基本的にZONE構成ノードは参加命令に対して忠実に参加開始すればよい。ただ例外として、ZONE併合処理が前後した場合に対処するために、自身が既に併合先ZONEに所属しているかをチェックして処理を開始する。
次に、参加処理中に新たな参加プロセスが動作しないように、参加処理中であることを示すフラグをオンにする。これは、参加処理が終了するときにオフにされる。
最後に、参加メッセージを受信したノードの所属ZONEが変更されていた場合、参加処理を実行しているノードに再度ZONE参加処理依頼を送信してこのZONE参加処理を終了する。
ところでZONE参加フローの変更ポイントの第1点で指摘している場合について、その具体的状況を図5.10を例に説明しておく。図の示す内容は、ノードBはノードAの参加ZONEがDEFAULT_ZONEなのでノードAにZONE[α]参加要請を送信するが、ノードAが既に別ZONEに所属している状況である。

続いて、ZONE併合フローの変更ポイントは2点、自ZONE及び併合交渉先ZONEがZONE_IDを変更していないか、ZONE併合交渉中に他ノードがZONE併合交渉を開始する場合に対処できるかという点である。前者については、両ZONEについてZONE_IDの変更を監視する。自ZONEについては、ZONE_IDの変更を検知し、参加先が併合交渉先ZONEである場合は即併合交渉を終了する。そうでない場合はパケットに載せる自身のZONE情報を変更して処理を続行する。併合交渉先ZONEのZONE_IDが変更されていて、かつ参加先が自ZONEでなければ、併合交渉処理を再初期化する。後者については、併合交渉処理を開始するときにフラグをたてることで対処する。

5.3 ZONE情報の収集

本節では、ZONEスタビライズ処理と、他ZONE情報の収集方法について述べる。

5.3.1 ZONEスタビライズ

Zoneオーバレイ上のDEFAULT_ZONE以外に所属する各ノードは、一定期間ごとに自ZONEの更新情報を収集するためにZONEスタビライズを実行する。その処理フローを図5.13に示す。フローの解説については第4.3節を参照。

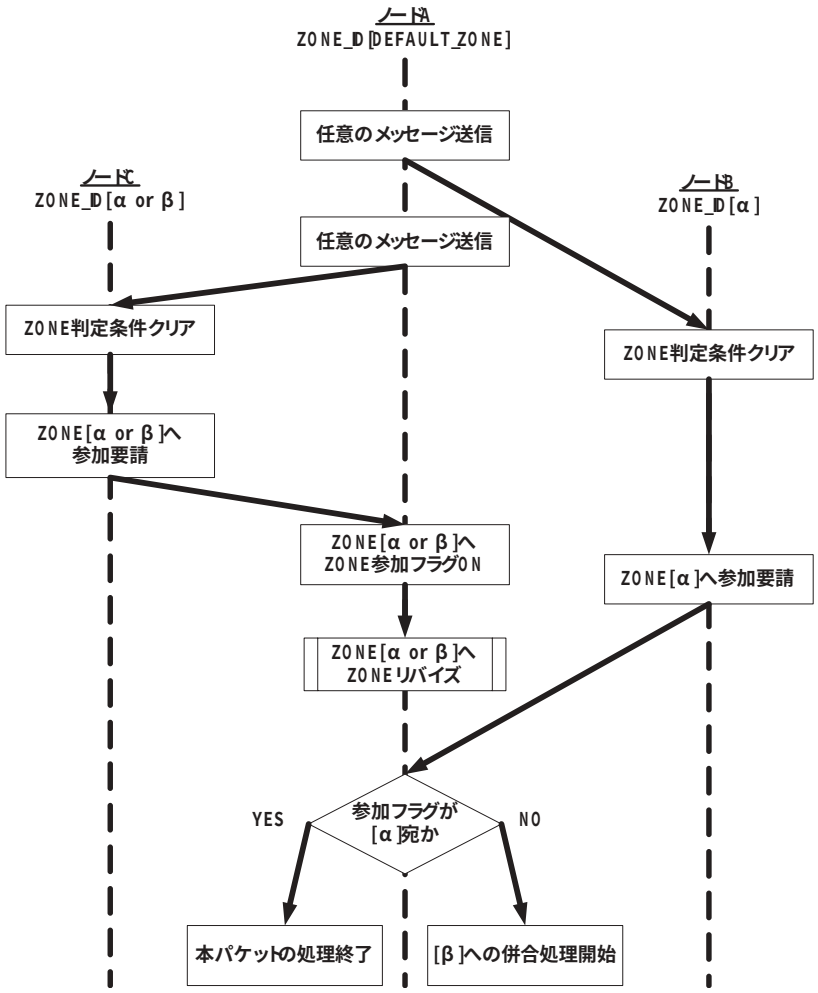

図 5.10: DEFAULT_ZONE ノードによる ZONE 参加処理の複雑化

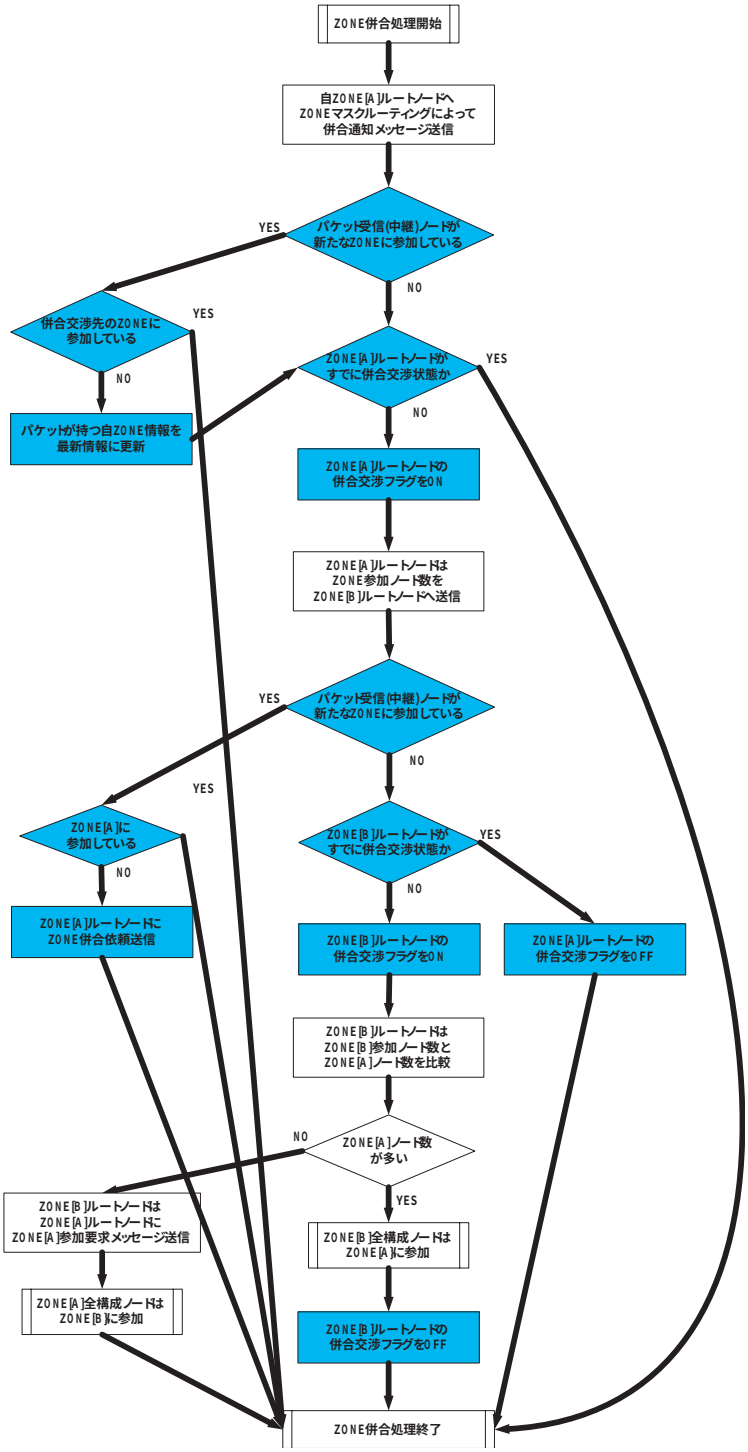

図 5.11: (改)ZONE 併合フロー

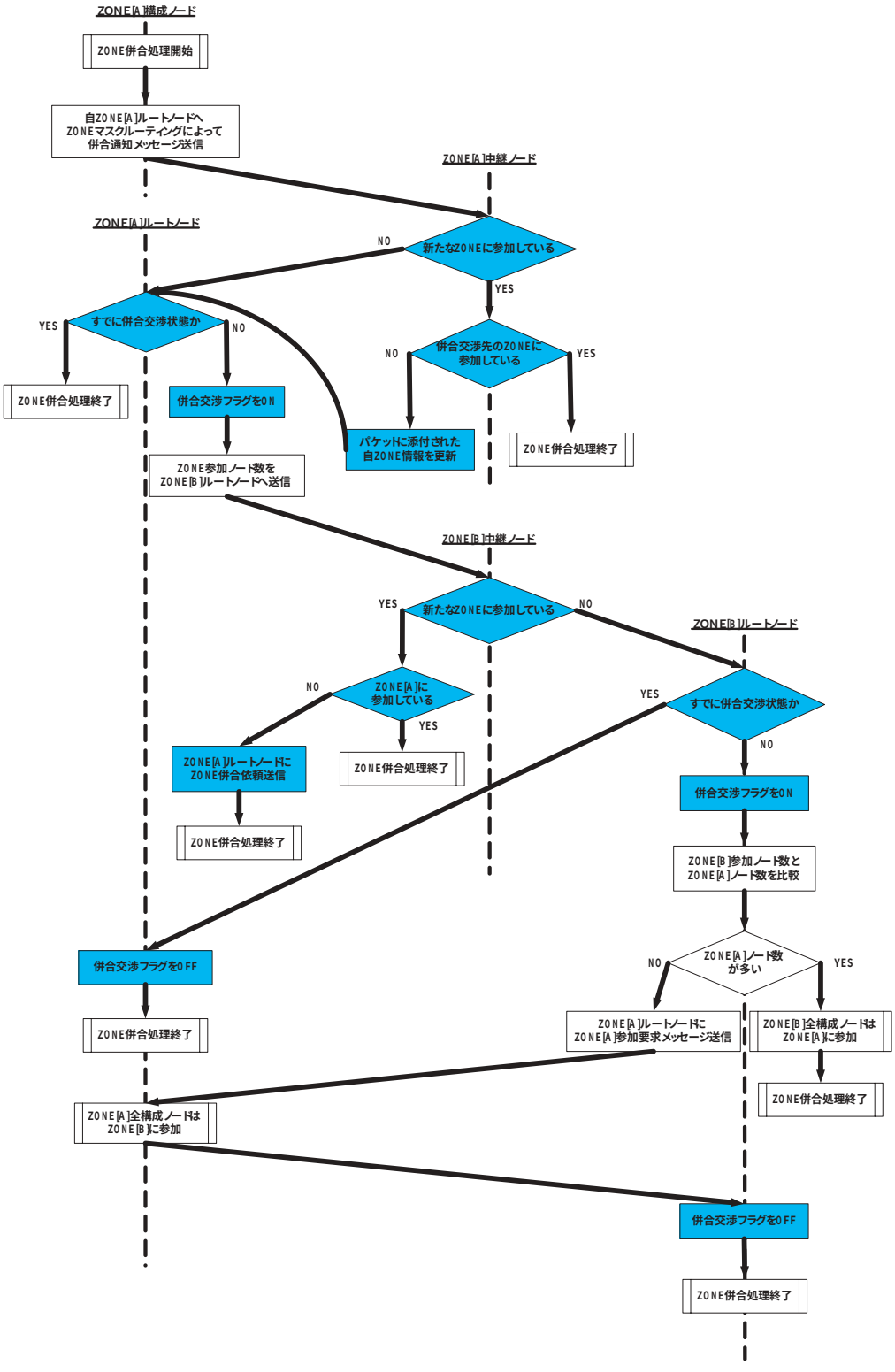

図 5.12: (改)ZONE 併合シークエンス

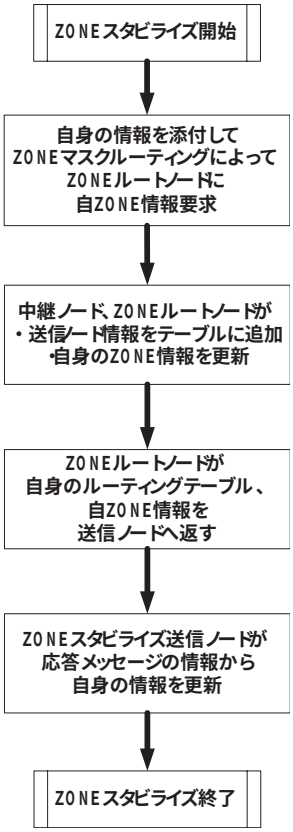

図 5.13: ZONE スタビライズフロー

5.3.2 他 ZONE 情報の収集

他 ZONE 情報の収集場面はオーバレイ上でコンテンツ配信に関するルーティング等が実行される全ての通信処理である。例えば Tapestry オーバレイにおけるスタビライズメッセージやクエリールーティング、パブリッシュルーティングなどが挙げられる。

5.4 ノードリバイズ対応 Tapestry ルーティング

第 4.4.1 節で述べた短所について Zone オーバレイでは、ZONE 所属変更処理が常により適切な方向に行われると仮定する。このときノードリバイズ処理の負荷は必ず収束する。例えば全ノードが適切な ZONE に所属した後は該当の負荷は出現しない。
またノードリバイズ対応の処理として、スタビライズ処理が挙げられる。スタビライズ処理の際に相手ノードの所属 ZONE が変更されていた場合に自身のルーティングテーブル該当エントリを削除する。

5.5 Zone Overlay Replication

各コンテンツのルートノードは一定期間毎にコンテンツの複製配置を実行する。具体的には図 5.14 のようなフローが実行される。コンテンツ配置候補 ZONE の判定条件としては、コンテンツが 1 つも存在しない ZONE やリクエスタからリクエストが到着した際にあらかじめリクエスタの ZONE を把握しておき、そういったリクエスタの多い ZONE を選択するなどが考えられる。

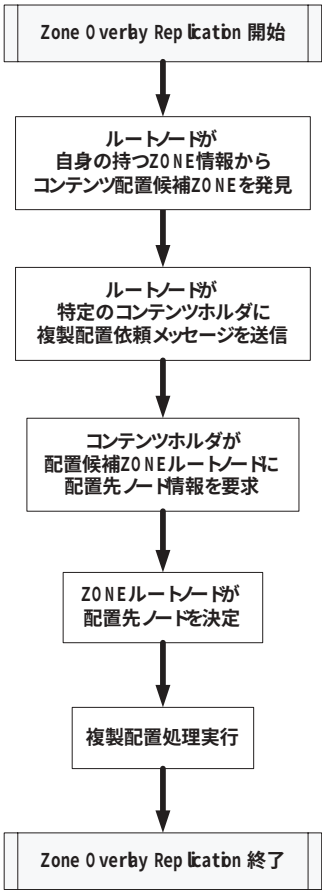

図 5.14: Zone Overlay Replication フロー

第6章　評価実験と考察

本章では、第 4 章、第 5 章で述べた Zone オーバレイについて実験評価を行い、考察を述べる。

6.1　実験環境

本研究の評価実験について、実験環境を以下に示す。

- ハッシュ関数は MD5 とし、GUID、NodeID ともに 16 進数 128bit(32 桁) の名前空間を用いる。
- GUID はコンテンツのタイトルをキーとしたハッシュ値 (128bit)、NodeID は、16byte の ZONE_ID 及び UID で構成される。UID は MAC アドレス (64bit) を元にして生成する。
- ルーティングテーブルの大きさを縦 32byte ×横 16byte とする。
- ソフトウェア上での実験用ネットワークとしては簡単なツリー型ネットワークが実装されている (図 6.1 参照)。
- ノードが ZONE を構成するための ZONE 判定条件は、ホップ数 2 とする。
- 障害などによってノードが突然オーバレイから脱退しない。

上記以外に、各実験で特に明記しない場合は以下の設定を用いる。

- 図 6.1 のネットワークの各パラメータについて、ルータ数は $K = 2$、$L = 2$、$M = 4$、ノード数は $N = 4, 8, 16, 32, 64, 128$ として実験、比較する。尚、ノード数をこのとき全参加ノード数は 64、128、256、512、1024 及び 2048 ノードである。
- 各ノードの参加間隔は 1 分とする。
- 参加処理の最初に参照するオーバレイ上のノードは原則固定ノードを用いる。

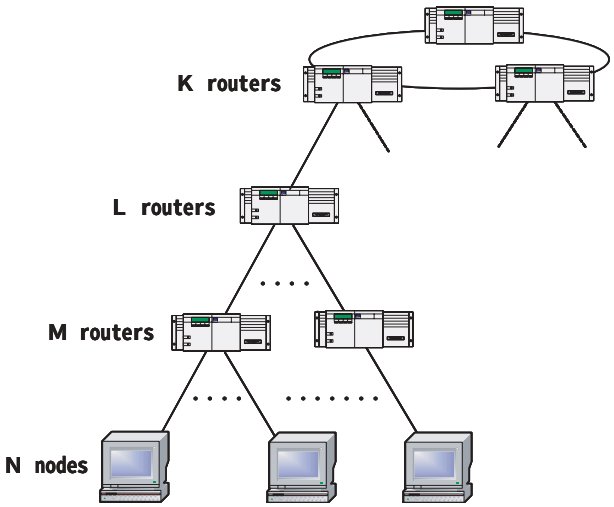

図 6.1: 実験用ツリー型ネットワーク

- 各ノードは 20 分に 1 回、ルーティングテーブル 1 エントリについてスタビライズ処理を行う。
- 各 ZONE は生成されてから ZONE スタビライズ処理を 20 時間に 1 回行う。
- コンテンツホルダのパブリッシュ処理の間隔を 12 時間とする。
- Zone Overlay Replication のコンテンツ配置先 ZONE の条件をコンテンツが 1 つも存在しない ZONE とする。

6.2 評価実験と考察

本節では ZONE の評価実験として、ノード参加処理が一度に多数行われる場合を想定し、そのときの ZONE の生成速度、完成度、情報収集率、ルーティングテーブル有効エントリ率への影響を解析する。これらの結果は、ノードがランダムなタイミングでオーバレイに参加する場合と比べて、多少ネットワークに与える影響は大きく、負荷実験の意味にもなる。上述した参加間隔のとき全ノードの参加終了時間は、64 ノードで約 1、128 ノードで約 2、256 ノードで約 4、512 ノードで約 8、1024 ノードで約 16、2048 ノードの場合で約 32 時間後である。
各評価実験で行った実験内容は大きく 6 通りに分けられる。各実験ではこれらのうち必要な実験を行っている。

実験 1 コンテンツなし (以降、C0 と示す)、1 日、各ノードは参加直後に自身に一番適した ZONE に所属する (以降、最適参加と表記)。具体的には参加時に参照するオーバレイ上のノードを同じ ZONE に所属すべきノードに変更する。

実験 2 C0、1 日、図 6.1 で示したネットワーク端に位置するノードから順番にオーバレイに参加する (以降、順列参加と表記)

実験 3 C0、3(もしくは 1) 日、ノードはランダムな順番でオーバレイに参加する (以降、ランダム参加と表記)

実験 4 コンテンツあり ((以降、C[X] と示す。[X] にはコンテンツ数が入る)、3 日、ランダム参加、Zone Overlay Replication を定期的に実行

実験 5 C[X]、3 日、ランダム参加、Zone Overlay Replication 実行なし

実験 6 C[X]、5 日、ランダム参加、Zone Overlay Replication 実行あり、クエリーあり

実験 1 は、最適な状況においての結果を参照するために行う。実験 2 は実験 3 のランダム参加の場合と比較するために行う。実験 3 は参加処理について評価するために行う。実験 4 はコンテンツパブリッシュメッセージや Zone Overlay Replication 処理といった、コンテンツ配信ネットワークの機能を評価するために行う。実験 5 はパブリッシュメッセージを増加させた場合を評価するために行う。実験 6 は、実験 4 のようにネットワークが機能する中でクエリー処理が起こることでより現実世界に近い状況を評価するために行う。
尚、実験 4、5 では、全ノード参加後にコンテンツをランダムなノードに投入し、その後コンテンツホルダは定期的にパブリッシュルーティングを送信する。実験 6 は、より現実に近い状況を想定して、頻繁に Zone Overlay Replication を実行し、かつクエリーも複数回実行する。ここでクエリーは、コンテンツ登録後数時間後に一定間隔でランダムなノードから送信する。

6.2.1 生成度評価

各ノードは何らかの ZONE に所属することで ZONE の効果を享受できる。そこで、ZONE 判定条件をクリアして ZONE に所属しているノードの割合を時系列グラフに示し、どの程度のノードが ZONE 効果を得るかを評価する。またこの際、1 ノードが ZONE に所属していく様子を表すため、現在オーバレイに参加しているノードを対象にした評価ではなく、シミュレーションで設定されたノード数を元にした割合として評価する。その評価式としては、ZONE 判定条件をクリアしたノードの数を計算し、その値をオーバレイに参加している、もしくは参加予定の全ノー

ド数で割ることで得られる。生成度 (%) は時間 t を変数とした以下の式で表すことができる (ZNNODE_CNT=DEFAULT_ZONE 以外の ZONE に所属するノード数、OVERLAY_NODE_CNT=シミュレーション設定ノード数)。

$$ZONE\,\text{生成度}_t = \frac{ZNNODE\_CNT}{OVERLAY\_NODE\_CNT} * 100$$
$$(t = \text{シミュレーションの任意時間})$$

グラフの見方は、ある時間においてグラフと X 軸に囲まれた範囲が ZONE 判定条件をクリアした経験のあるノードの割合を示す。

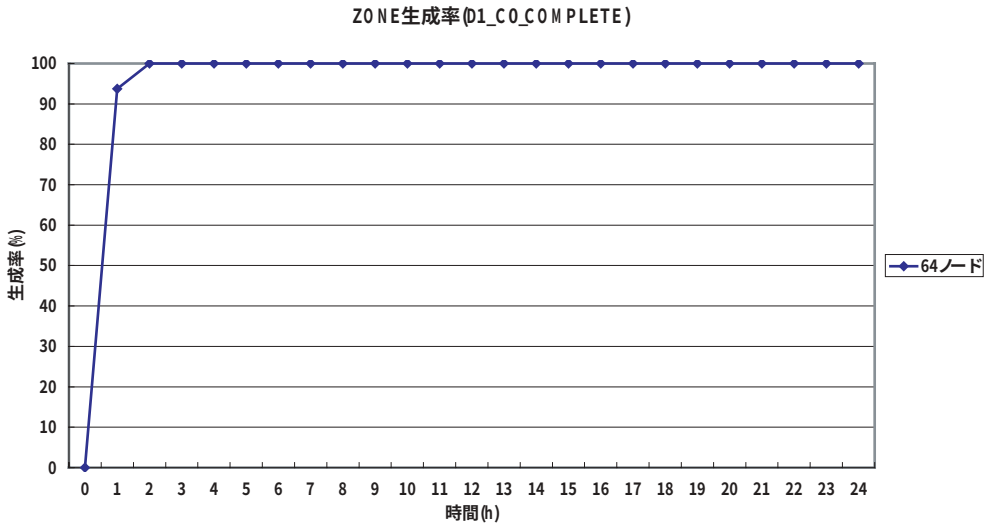

図 6.2: 最適参加

図 6.2(実験 1) は、ZONE 生成度向上の理想データとして 64 ノードの場合の結果を示している。2 時間後に全てのノードが参加し、かつ全ノードが参加処理で DEFAULT_ZONE 以外の ZONE に参加していることが分かる。

次に、図 6.3(実験 2)、6.4(実験 3) から、順列参加とランダム参加の場合を比較する。順列参加の場合は最適参加と同様に、全ノード参加終了時に何らかの ZONE に所属しているのに対し、ランダム参加の場合は、参加処理完了後にも DEFAULT_ZONE に所属するノードが存在することが分かる。

また、順列参加の場合は参加ノードが次々に ZONE 参加しているのに対して、ランダム参加ではそれに限らない。前者の場合を詳述すると、偶数番のノードがオーバレイ参加する度に ZONE が生成されている。具体的には、偶数番のノードが参加す

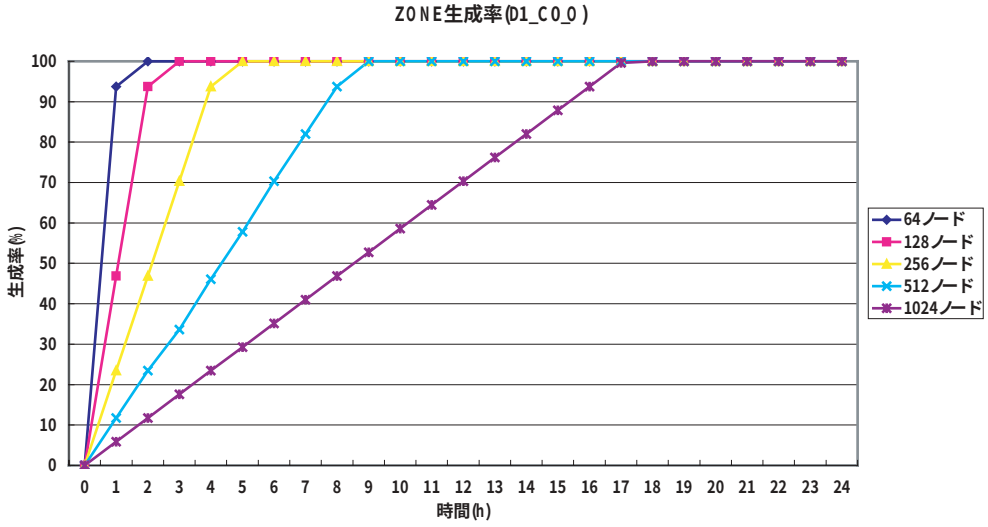

図 6.3: 順列参加

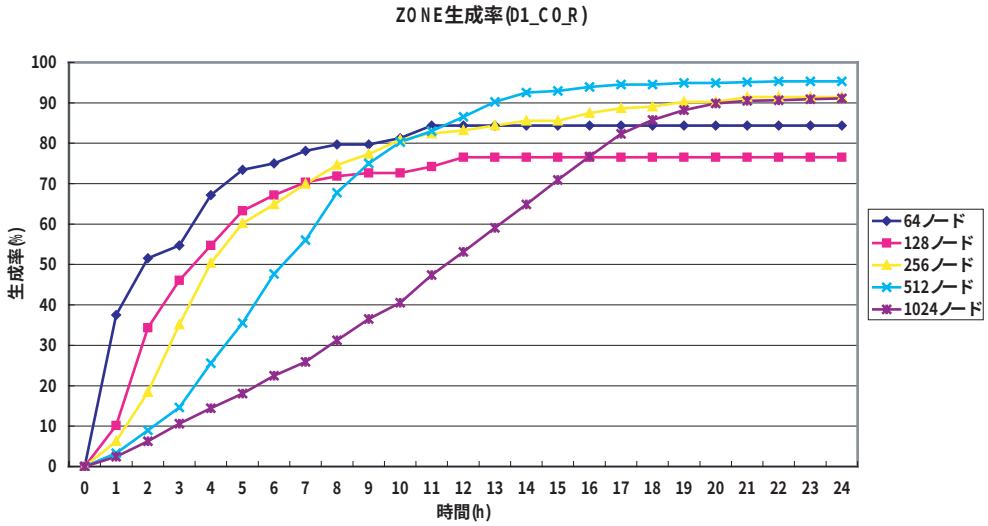

図 6.4: ランダム参加・C0・1 日

るときには必ず、直前に参加したノードのみが DEFAULT_ZONE に所属しているためこれをオーバレイ参加のための Tapestry ルーティングにて発見し、ZONE 生成が生成されている。この場合、各ノードは最適な ZONE に参加しているわけではないので、例えば次節で示す別評価では最適参加とは異なる評価となる。後者の場合は、オーバレイ参加処理の際に ZONE 参加処理を行わない場合があるためこのようなグラフになる。

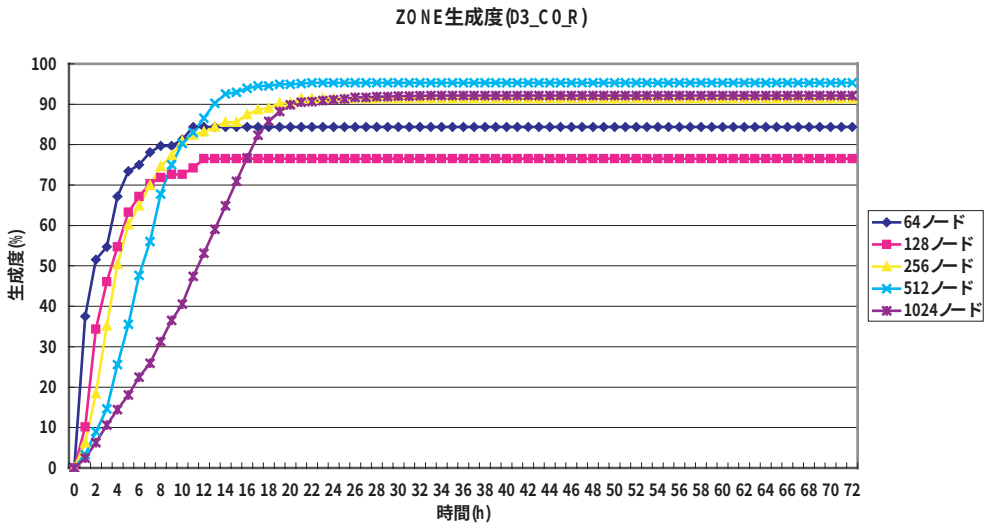

図 6.5: ランダム参加・C0・3 日

図 6.5 は図 6.4 と同様のシミュレーションを 3 日続けた場合であり、各 ZONE 生成度は全ノード参加後も上昇し、その後一定になる。図 6.4 と図 6.5 の場合の最終値を比較した表 6.1 より、参加してから 24 時間を過ぎてから値がほぼ変化していないことが分かる。

本実験において ZONE 生成度上昇の場面は、ノード参加処理とルーティングテーブルエントリのスタビライズ処理である。前者については、ノード間で参加処理に関わるパケットが授受されることにより、後者については、ノードが参加時にすでに参加しているノードのテーブルをコピーして構築したルーティングテーブルエントリのスタビライズ処理により ZONE を生成する。実験結果において 100%の評価を得られない理由は、各ノードのルーティングテーブルにオーバレイ上の全ノード情報が存在しないからである。これは、例えばコンテンツ検索で Tapestry ルーティングを行うときに、クエリ生成ノードがコンテンツホルダの情報を得る、もしくはルーティング経路上のノードがクエリ生成ノードの情報を得ることで向上できると考えられる。

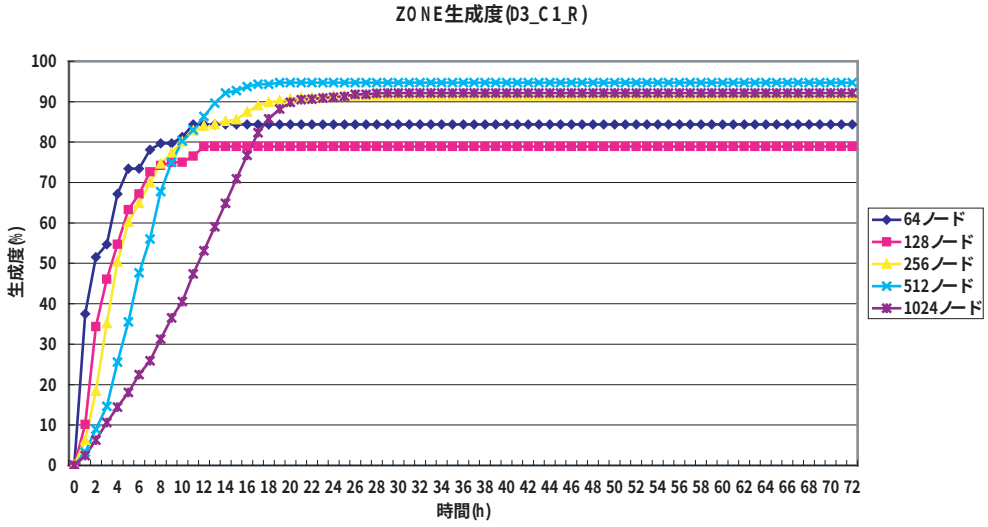

図 6.6: ランダム参加・C1・3 日

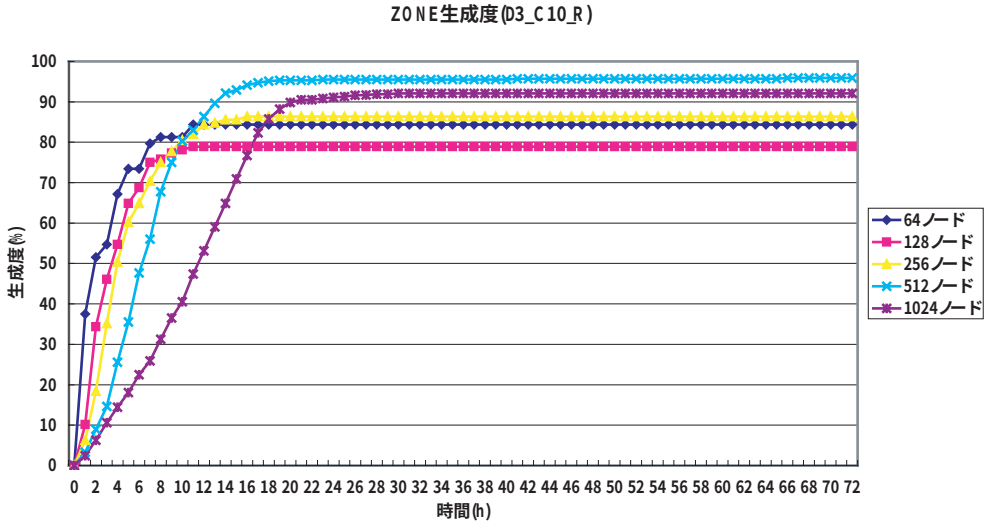

図 6.7: ランダム参加・C10・3 日

実験 No.	最終状態 平均値
C0・1 日 (図 6.4)	87.8%
C0・3 日 (図 6.5)	88.0%
C1・3 日 (図 6.6)	88.3%
C10・3 日 (図 6.7)	87.5%
C1・5 日・10 クエリー (図 6.8)	89.6%

表 6.1: ZONE 生成度 最終状態 平均値比較 (C0-1 日、C0-3 日、C1-3 日、C10-3 日、C1-5 日-Q10)

図 6.6(実験 4) は全ノード参加後に 1 コンテンツを投入し、20 時間ごとに Zone Overlay Replication を実行した場合の評価、図 6.7(実験 5) は全ノード参加後に 10 コンテンツを投入し、Zone Overlay Replication を実行しない場合の評価である。これらは図 6.5 の評価と似た結果が得られており、パブッリシュメッセージや Zone Overlay Replication が ZONE 生成度に与える影響は少ない。表 6.1 からもこのことが分かる。

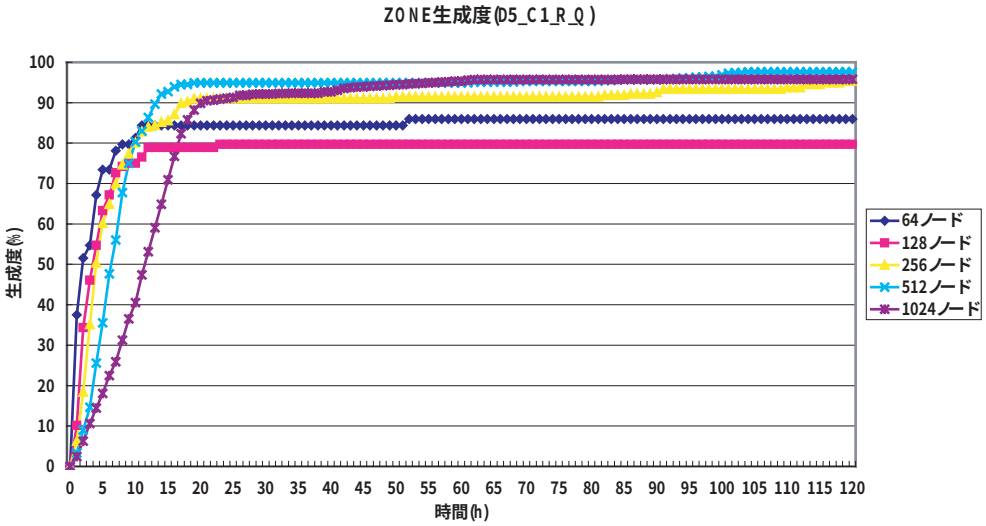

図 6.8: ランダム参加・C1・5 日・クエリー 10

図 6.8(実験 6) は全ノード参加後に 1 コンテンツを投入し、その後 Zone Overlay Replication が行われる中でクエリーを 10 回実行した場合の評価である。クエリー処理を実行することで ZONE が生成されることが分かる。表 6.1 からも他の場合と比べて平均値が上昇することが分かる。

6.2.2 完成度評価

本節では、前節で示した評価との比較として、どれだけのノードが適切な ZONE に分類されるかを解析する。具体的には下記に示す式によって、全ノードを適切な ZONE に分類した状態を基準にシミュレーション中の ZONE 状態の評価を行い、時系列グラフに示して比較する。評価値については、1 ノードが適切な ZONE に所属する場合に 1 点とし、また本来 1 つの ZONE であるノード集合に複数 ZONE が存在する場合は、ノード数が多い ZONE を適切な ZONE と見なす。ここで ZONE 生成度評価と同様に、1 ノードが適切な ZONE に所属していく様子を表すために割合を表す元の値をシミュレーションで設定したノード数にする。完成度 (%) は時間 t を変数とした以下の式で表すことができる (ZNNODE_CNT=ZONE 所属ノード数、OVERLAY_NODE_CNT=シミュレーション設定ノード数)。

$$ZONE\text{完成度}_t = \frac{\sum^{\text{適切な } ZONE \text{ 数}} ZNNODE\_CNT}{OVERLAY\_NODE\_CNT} * 100$$
$$(t = \text{シミュレーションの任意時間})$$

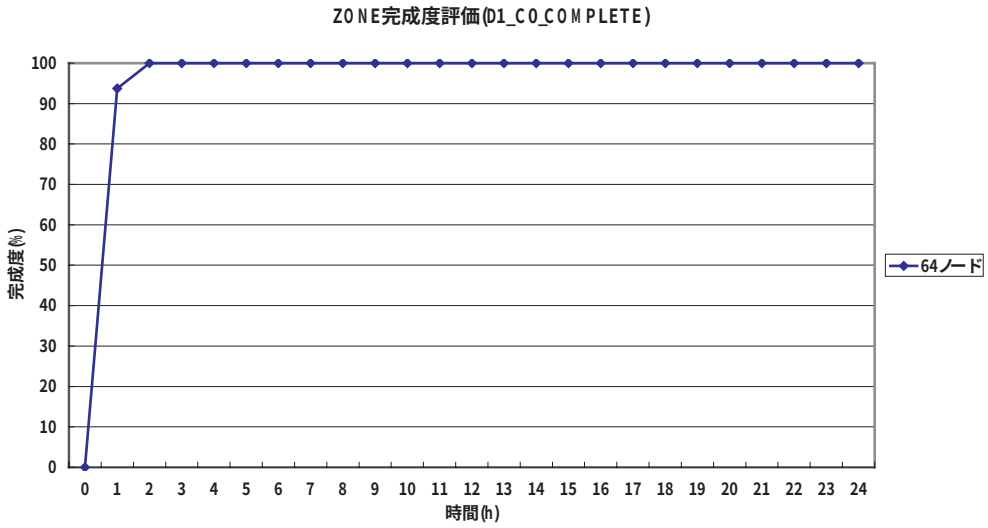

図 6.9: 最適参加

図 6.9(実験 1) は、ZONE 完成度向上の理想データとして 64 ノードの場合の結果を示している。前節の場合と同様に、全ノード参加後に全ノードが適切な ZONE に所属していることが分かる。

図 6.10(実験 2)、6.11(実験 3) は、それぞれ順列参加、ランダム参加の場合の ZONE 完成度を示している。前者の場合は、前節で示したように ZONE 生成度は最適参加

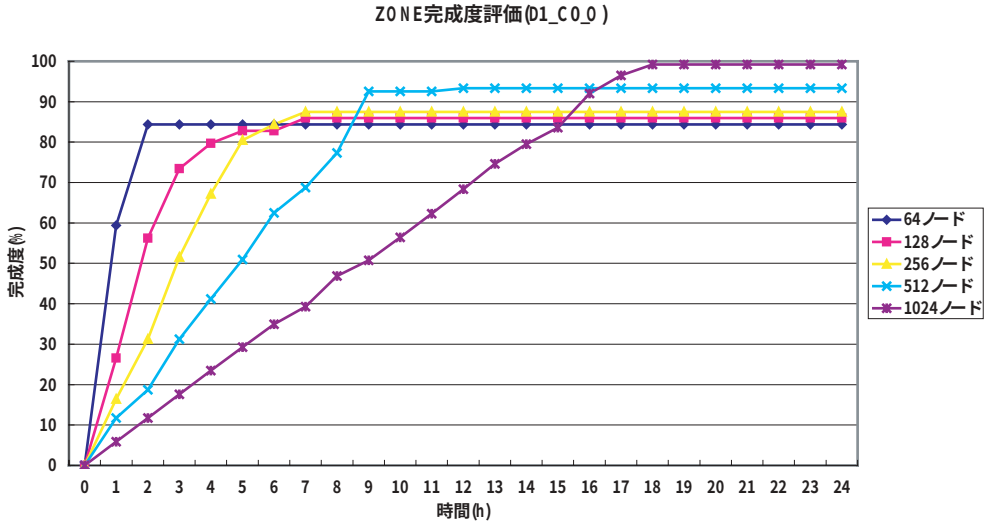

図 6.10: 順列参加

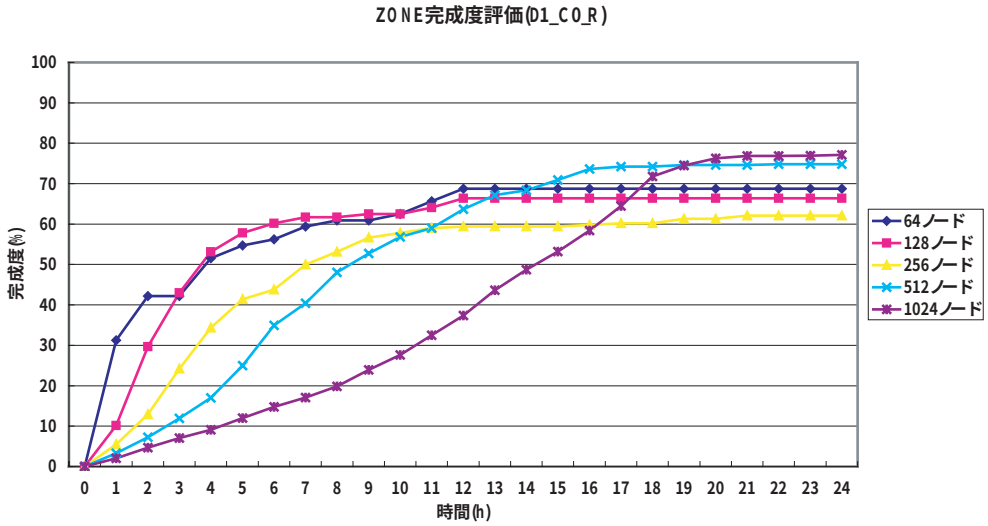

図 6.11: ランダム参加・C0・1 日

図 No.	1 日後 平均値
順列参加 (図 6.10)	90.1%
ランダム参加 (図 6.11)	69.8%

表 6.2: ZONE 完成度 1 日後 平均値比較 (順列参加-ランダム参加)

の場合と同様の結果が得られたが、このグラフより生成された各 ZONE は併合されない場合があることが分かる。特徴として、ノード数の増加とともに最終的な完成度が上昇すること、1024 ノードでは約 100%になることが挙げられる。後者については、ZONE 完成度の ZONE 生成度に示す割合を別図 6.12 に示す。グラフから、生成された ZONE のうち適切な ZONE がどれだけ存在するかを測れる。この結果、生成された ZONE のうち多くが適切な ZONE であることが分かる。

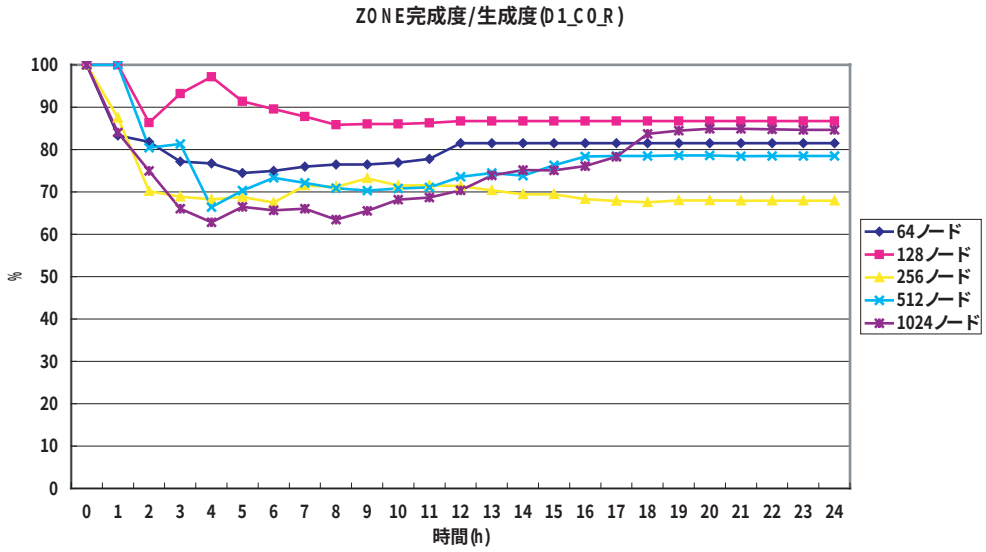

図 6.12: 完成度が生成度に占める割合、ランダム参加・C0・1 日

また、図 6.10 と図 6.11 の 1 日後の平均値を比較した表 6.2 では、ランダム参加の場合に 20%も低いことが分かる。

図 6.13 は図 6.11 と同様のシミュレーションを 3 日続けた場合である。図 6.11 と図 6.13 の場合の最終状態の平均値を比較した表 6.3 より、参加してから日時が経過しても ZONE 完成度はほぼ上昇しないことが分かる。

図 6.14(実験 4) は、全ノード参加後に 1 コンテンツを投入し、その後 Zone Overlay Replication を実行した場合の評価、図 6.15(実験 5) は、全ノード参加後に 10 コンテンツを投入した場合の評価であるが、前節と同様に Zone Overlay Replication 処理や

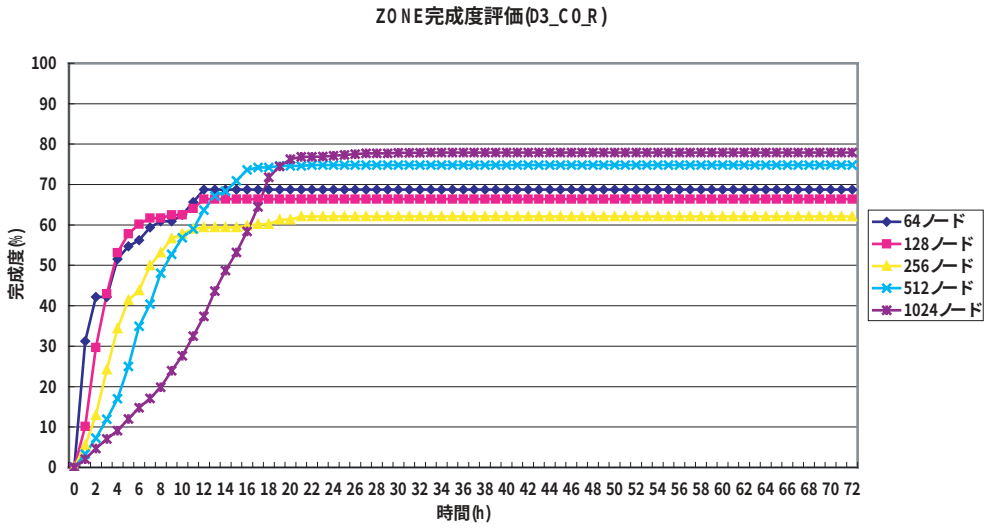

図 6.13: ランダム参加・C0・3 日

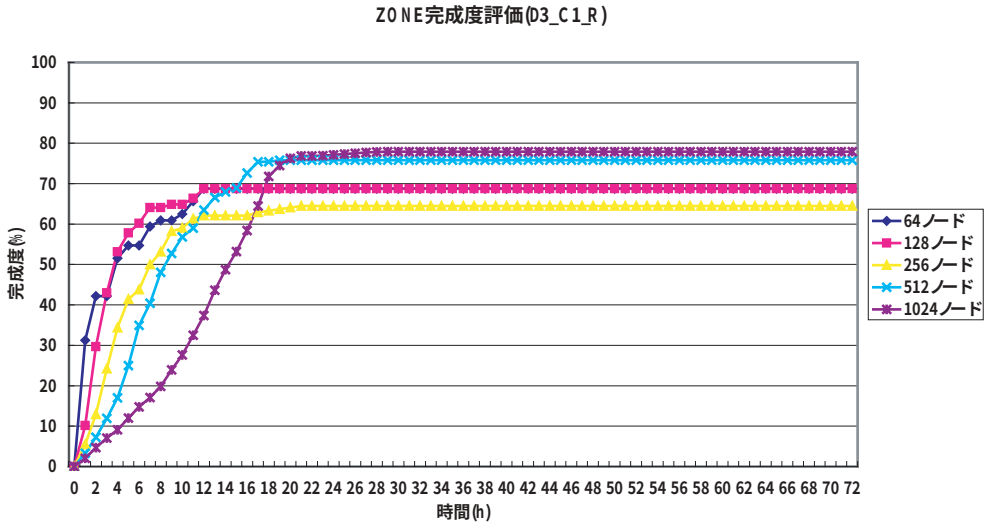

図 6.14: ランダム参加・C1・3 日

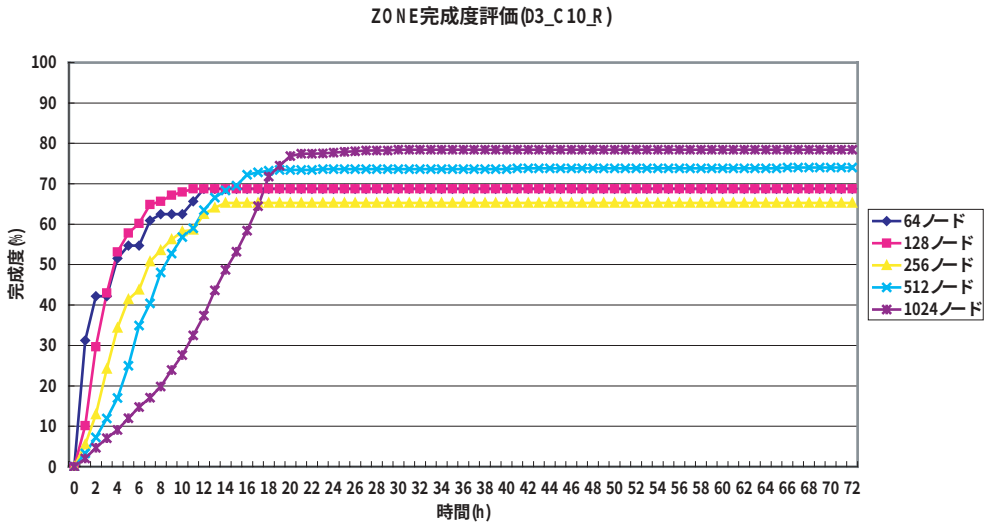

図 6.15: ランダム参加・C10・3 日

実験 No.	最終状態 平均値
ランダム参加・C0・1 日 (図 6.11)	69.8%
ランダム参加・C0・3 日 (図 6.13)	70.0%
C1・3 日 (図 6.14)	71.1%
C10・3 日 (図 6.15)	71.0%
C1・5 日・10 クエリー (図 6.16)	74.1%

表 6.3: ZONE 完成度 最終状態 平均値比較 (C0-1 日、C0-3 日、C1-3 日、C10-3 日、C1-5 日-Q10)

コンテンツパブリッシュが ZONE 完成度に与える影響は少ない。表 6.3 からもこれが分かる。

図 6.16(実験 6) の実験内容は前節で示したものと同様である。グラフより、クエリー処理によって ZONE 完成度が向上していることが分かる。表 6.3 からも他の場合と比べて平均値が向上している。この理由は、Zone Overlay Replication によって各 ZONE に分散配置されたコンテンツホルダがクエリー応答をリクエスタに返す際に ZONE 併合処理が起きているためと考えられる。

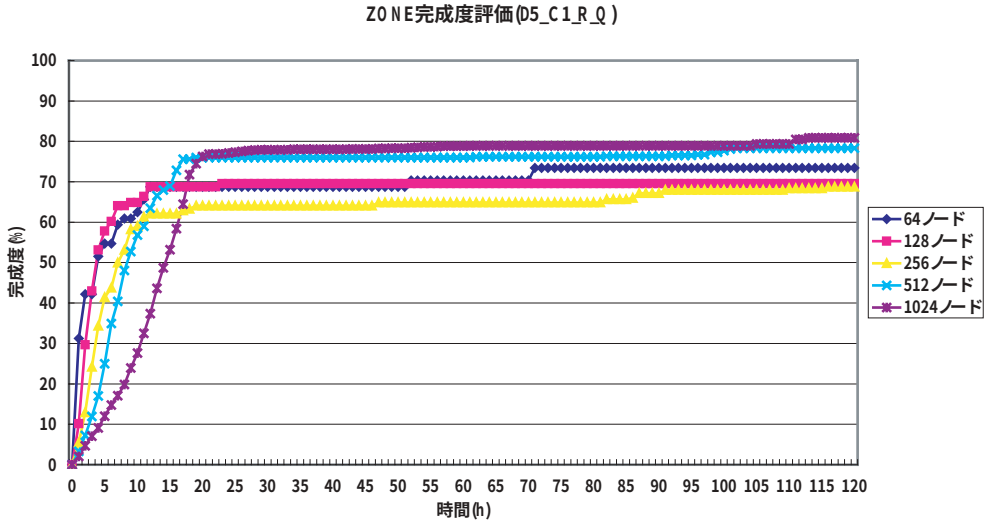

図 6.16: ランダム参加・C1・5 日・クエリー 10

6.2.3 情報収集率評価

ノードが適切な ZONE に所属しても、オーバレイ上の全ノードがその情報を収集しなくては ZONE を生かせない。また、収集した ZONE 情報は、それが併合されることで無効な情報となってしまう。そこで各ノードがどれだけ有効な ZONE 情報を保持するかを時系列グラフに出力して評価、比較する。評価値については、各ノードについて有効な ZONE 情報を 1 つ保持すると 1 点加算する。収集率 (%) は時間 t を変数とした以下の式で表すことができる ($NODE\_AVE$=全ノードの平均値、CUR_ZONE=ノードの持つ有効 ZONE 情報数、$TOTAL\_ZONE$=ノードの持つ ZONE 情報数)。

$$ZONE\text{ 情報収集率}_t = NODE\_AVE\left(\frac{CUR\_ZONE}{TOTAL\_ZONE}\right) * 100$$
$$(t = \text{シミュレーションの任意時間})$$

まず本節で示される各グラフを ZONE 完成度グラフを参照して総括すると、ZONE 完成度評価値が安定してから数時間以内に各ノードの持つ ZONE 情報が最新状態に更新されていることが分かる。ここで、ノードは 1 分間隔でオーバレイに参加し、かつ ZONE スタビライズ処理は各ノードが 20 時間に 1 回行う設定なので、全ノードが ZONE スタビライズ処理を行う前に ZONE 情報収集率は 100%を示している。つまり、テーブルエントリのスタビライズ処理時に ZONE 情報を交換することで、ZONE スタビライズ処理時に得られる自 ZONE 情報をカバーする場合が存在することを示

している。
また ZONE 収集率はシミュレーション中の ZONE 生成処理に依存するグラフで、ZONE 生成が行われることで一時的に ZONE 収集率が下がる可能性がある。例えばZONE が生成された瞬間、ZONE に無関係のノードはその ZONE 情報を保持していないので、よって ZONE 収集率は下がる。

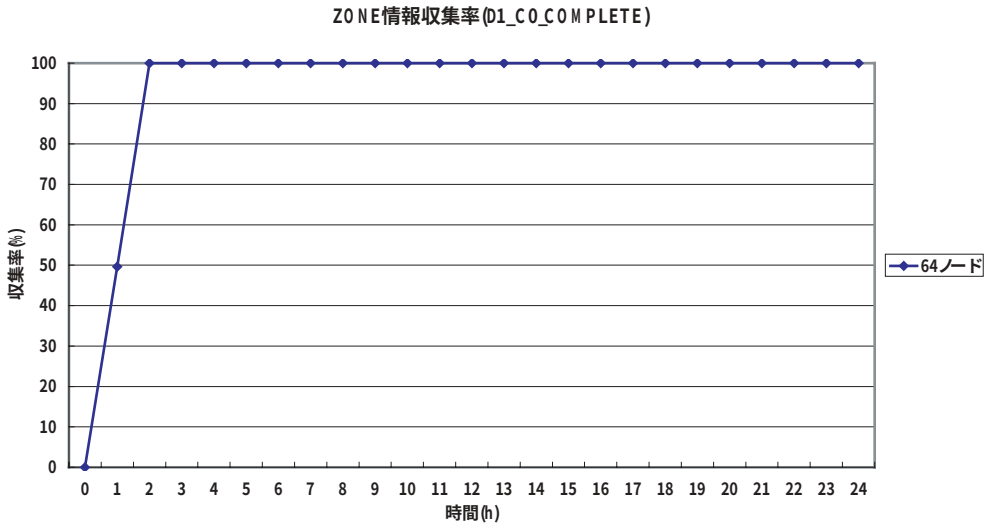

図 6.17: 最適参加

図 6.17(実験 1) は、前節と同様に理想データとして 64 ノードの場合の結果を示している。理想的な状況では、ノードの参加処理が終了 (全ノードが適切な ZONE に所属) してから 1 時間以内の間に全ノードがその情報を収集している。

図 6.18(実験 2)、6.19(実験 3) は、それぞれ順列参加、ランダム参加の場合の ZONE 情報収集率を示している。両者の相違点は、各ノードが ZONE 情報を収集するまでの時間が順列参加の方が短い点。また、両実験から、オーバレイ上に存在する ZONE 情報の収集という点については、極めて早い段階で 100%に近い成果が得られることが分かる。

図 6.20 は図 6.19 と同様のシミュレーションを 3 日続けた場合である。ZONE 生成が行われなくなってから遅くても 20 時間以内に ZONE 情報収集率は 100%となることが分かる。

以上より、ZONE 情報収集率についてはスタビライズ処理・ZONE スタビライズ処理によってすぐに 100%の成果を得られることが実証できた。

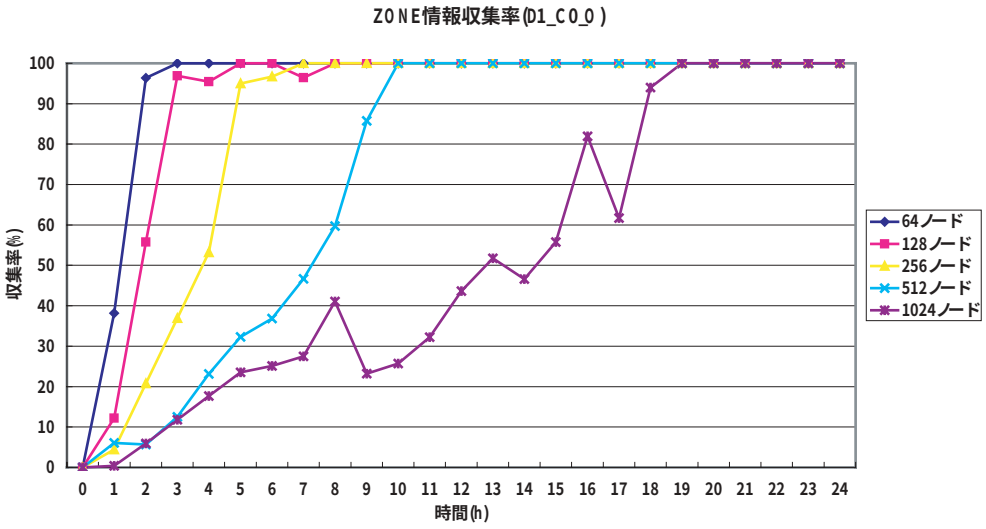

図 6.18: 順列参加

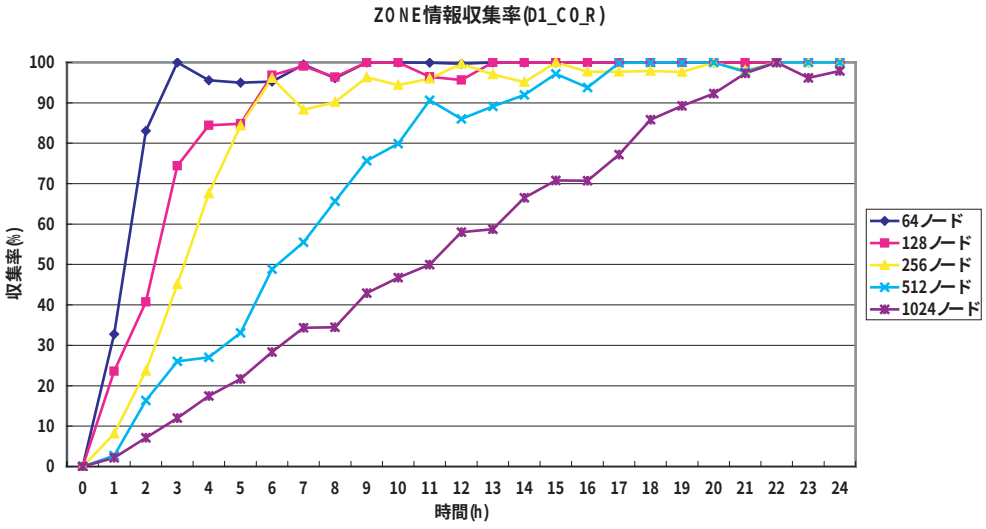

図 6.19: ランダム参加・C0・1 日

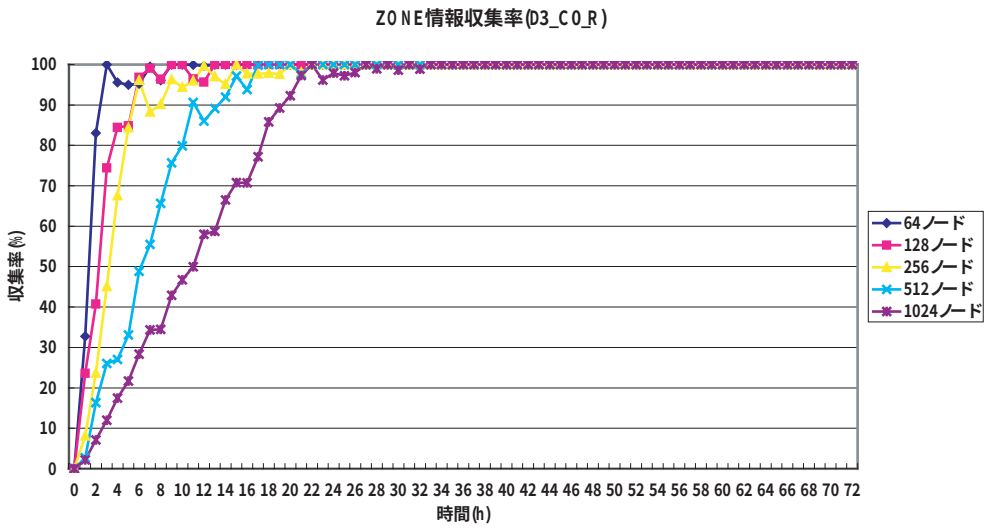

図 6.20: ランダム参加・C0・3 日

6.2.4 ノードリバイズによる Tapestry オーバレイ完成度評価

Zone オーバレイでは各ノードは ZONE が変更される度にノードリバイズ処理によってオーバレイに再参加を繰返すので、それによって全ノードのルーティングテーブル (以降本節では、RT と表記) の該当エントリが無効になる。無効エントリ数が増大すると Tapestry ルーティングの精度等に影響を与える可能性が考えられる。そこで本節では、各ノードの RT 有効エントリの割合を解析することで Tapestry オーバレイ完成度 (%) を評価する。RT 有効エントリ率は時間 t を変数とした以下の式で表すことができる ($NODE\_AVE$=全ノードの平均値、$CUR\_RTENTRY$=RT 有効エントリ数、$TOTAL\_RTENTRY$=RT 全エントリ数)。

$$RT\text{ 有効エントリ率}_t = NODE\_AVE\left(\frac{CUR\_RTENTRY}{TOTAL\_RTENTRY}\right) * 100$$
$$(t = \text{シミュレーションの任意時間})$$

図 6.21(実験 1) は前節と同様に理想データとして 64 ノードの場合の結果を示している。全ノードが適切な ZONE に所属してから 1 時間以内に、全ノードの RT 有効エントリ率はほぼ 100%になっていることが分かる。
この実験のように全ノードが短時間に所属 ZONE を変更すると、一旦は RT 有効エントリ率は評価を落としてしまうが、他ノードの参加処理における Tapestry ルーティングや RT エントリのスタビライズ処理時にノードの所属 ZONE 変更を検知するこ

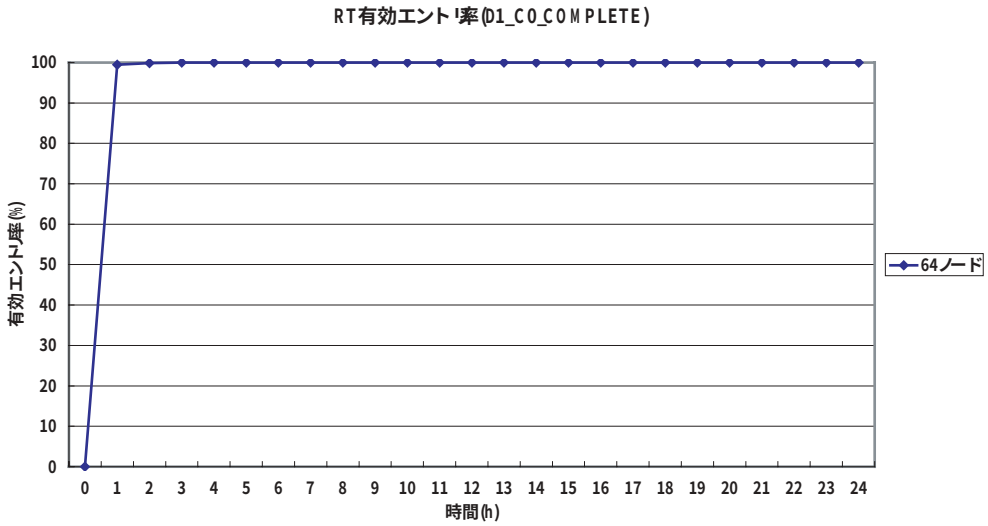

図 6.21: 最適参加

とでこれを修正している。

図 6.22(実験 2)、6.23(実験 3) は、それぞれ順列参加、ランダム参加の場合の評価である。グラフから、両者ともに 90%後半の値を続け、全ノードの参加処理が終了すると 100%に収束していくことが分かる。

図 6.24 は図 6.23 と同様のシミュレーションを 3 日続けた場合である。結果より、シミュレーション全ノード参加後数時間で RT 有効エントリ率は 100%になることが分かる。

6.2.5 評価実験を通しての考察

第 6.2.1 節、第 6.2.2 節で示した ZONE 生成度及び完成度については値が理想値よりも低く、改善の余地があると考えられる。だがその中で、Zone Overlay Replication によって配置されたコンテンツにクエリーすることによって、短期間に ZONE 完成度を向上できたのは評価できる。第??節で示した ZONE 情報収集率については短期間で 100%の値を得るため、オーバレイ上に存在する全 ZONE 情報をすぐに活用できる仕組みであると考えられる。また、第 6.2.4 節で示したようにノードリバイズによるルーティングテーブルへの影響については、短期間に小さい影響が出る程度であることを示した。この実験結果に加えて、Tapestry はノードの参加や脱退に柔軟に対応できる仕様となっているので、Zone オーバレイが Tapestry ルーティングの精度に与える影響は僅かであると考えられる。

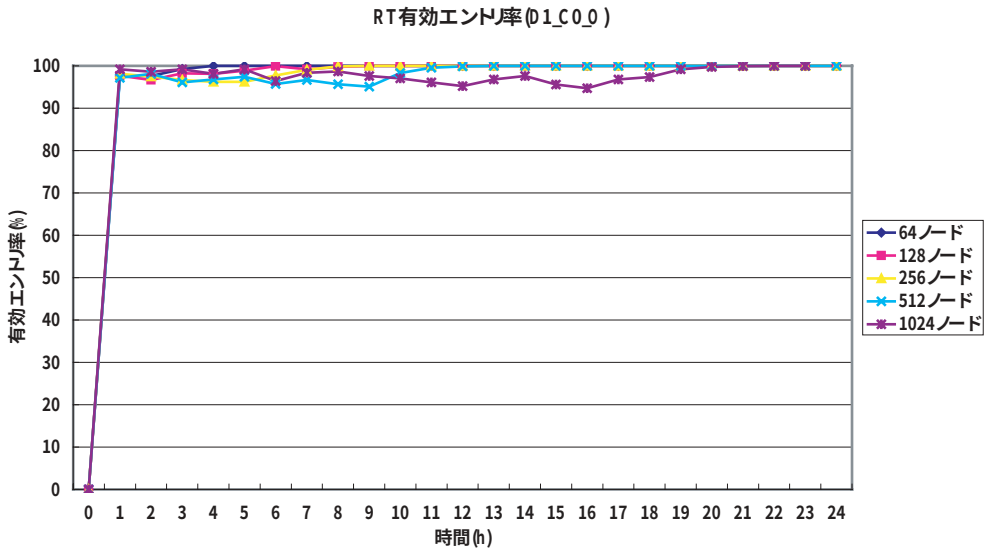

図 6.22: 順列参加

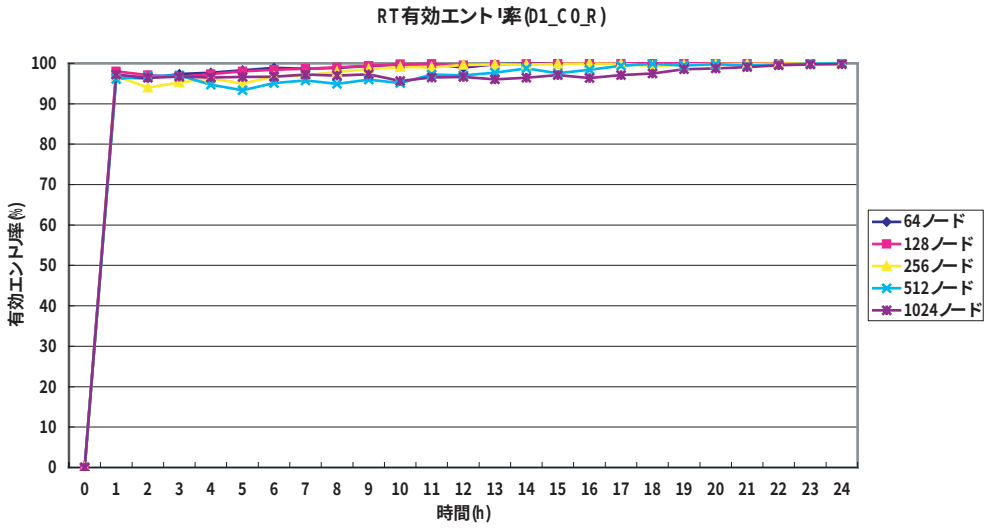

図 6.23: ランダム参加・C0・1 日

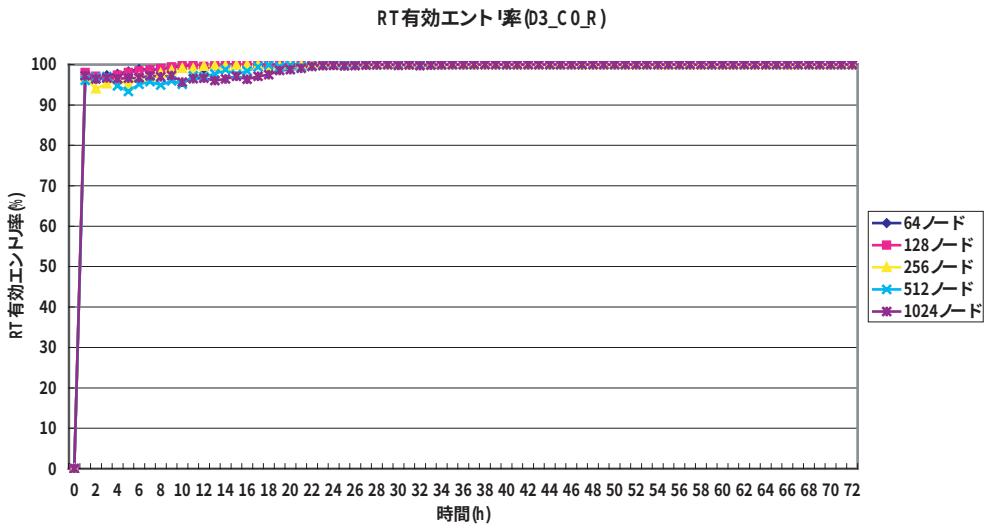

図 6.24: ランダム参加・C0・3 日

6.3 Zone オーバレイ仕様についての考察

6.3.1 RTT による ZONE 判定条件

本稿で示した実験では ZONE 判定条件としてホップ数を用いたが、実際のネットワークでは、例えば幾つものサブネットを用意する場合などにノード間の距離を厳密に測れない。そのような場合にノード間の距離を計測する方法として RTT(Round Trip Time) などを利用する方法もある。RTT を用いる場合は、pathchar [20] 等などの研究から、計測結果に差が大きいことが知られているので、例えば何度か RTT を計測した中で最悪な RRT 値を使用することで、各ノード間の距離の標準値として用いることができるのではないかと考える。

6.3.2 ZONE 生成時のクエリー集中

全ノードが DEFAULT_ZONE に所属しているときに ZONE が生成されると、クエリーなど Tapestry ルーティングメッセージの経路が ZONE_ID に集中する、という事態が考えられる。それに対して DEFAULT_ZONE をあらかじめ複数用意することで、ある程度回避できると考える。
例えば ID 空間が {0000}〜{EEEE} だとしたら、これを均等にするように DEFAULT_ZONE を {0000}、{1111}、{2222} … {EEEE} と 16 個設定しておき、オーバレイに参加す

るノードはランダムにどこかの DEFAULZ_ZONE に所属するようにする。

6.3.3 コンテンツ配置の冗長性

Zone オーバレイ仕様の特徴から、Zone Overlay Replication の目標が例えば各 ZONE に 1 コンテンツを配置する場合などでは本手法によって冗長にコンテンツを配置してしまう可能性がある。これは、ZONE は完成度が 100%になるまで非同期で生成、併合され続けることが理由で、例えばある時間において各 ZONE に 1 コンテンツを配置されていてもその ZONE が併合されることで 2 コンテンツ存在する ZONE が出現する。

第7章　結論

本章では、本稿の結論を述べる。

7.1　まとめ

本研究では、ノードの地理属性を考慮したZoneオーバレイを構築し、その上でそれを応用したZone Overlay Replicationを実装した。これによって、従来のTapestry-DHTと比べてノードの実ネットワーク上の位置を把握できるため、より戦略的な自律動作を実現できる。
実験を通して以下のZoneDHTの評価を示した。

- ZONE生成度及び完成度については改善の余地があるが、その中でZone Overlay Replicationによって配置されたコンテンツにクエリーすることによって、短期間にZONE完成度を向上できた。
- 短期間で100%の値を得るため、オーバレイ上に存在する全ZONE情報をすぐに活用できる仕組みである。
- ZONEを生成することによるTapestryルーティングテーブルエントリへの影響は非常に小さいので、提案したZoneDHTを使用することで与えるTapestryルーティング精度への影響は少ない。

7.2　今後の課題

今後の課題としては、ZONE生成度及び完成度を向上させる施策やより効率的なZONE判定条件の検討、ZONE生成時のクエリー集中を避ける機構の提案、コンテンツ配置の冗長性に対する対応などが挙げられる。

関連図書

[1] Ben Y. Zhao, Ling Huang, Jeremy Striblong, Sean C. Rhea, Anthony D. Joseph, Member, IEEE, and John D. Kubiatowicz, Member, IEEE. Tapestry: A Resilient Global-scale Overlay for Service Deployment. IEEE Journal on selected areas in communications, vol.22, No.1, January 2004.

[2] Antony Rowstron, Peter Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany, pages 329-350, November 2001.

[3] John Kubiatowicz, David Bindel, Yan Chen, Steven Czerwinski, etc. OceanStore: An Architectture for global-Scale Persistent Storage. In Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000), November, 2000.

[4] Ben Y. Zhao, John D. Kubiatowicz and Anthony D. Joseph. Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing. Report No. UCB/CSD-01-1141 April 2001.

[5] Yan Chen, Lili Qiu, Weiyu Chen, Luan Nguyen and Randy H. Katz. Clustering Web Content for Efficient Replication. Proceedings of the 10th IEEE International Conference on Network Protocols (ICNP'02).

[6] Michael J, Freedman and David Mazieres. Sloppy hashing and self-organizing clusters

[7] Michael J, Freedman, Eric Freudenthal, David Mazieres. Democratizing content publication with Coral

[8] Ian Clarke, Oskar Sandberg, Rrandon Wiley, and Theodore W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System.[1] In Proc. International Workshop on Design Issues in Anonymity and Unobservability, volume 2009 of LNCS, pages 46-66. Springer-Verlag, 2001.

[9] 中内 清秀, 森川 博之, 青山 友紀. 関連性を有するコンテンツのための分散ストレージサービス. 信学技法 TECHNICAL REPORT OF IEICE.

[1] the most cited computer science paper of 2000 according to Citeseer.

[10] Sylvia Ratnasamy, Sccott Shenker, Ion Stoica. Routing Algorithms for DHTs: Some Open Questions. Electronic Proceedings for the 1st International Workshop on Peer-to-Peer Systems (IPTPS '02).

[11] Hari Balakrishnan, M.Frans Kaashoek,David Karger,Robert Morris, and Ion Stoica (訳者：金田憲二). LOOKING UP DATA in P2P Systems(P2P システムにおけるデータの検索). Communications of the ACM, Vol.46, No.2, February 2003, pp. 43-48.

[12] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, Hari Balakrishnany. Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications. SIGCOMM '01, August 27-31, 2001, San Diego, California, USA.

[13] Jeffrey Considine. Cluster-based Optimizations for Distributed Hash Tables. Computer Science Department Boston University

[14] Akamai Technologies, Inc.
http://www.akamai.com/index_flash.html

[15] OceanStore Project
http://oceanstore.cs.berkeley.edu/

[16] Freenet Project
http://freenetproject.org/

[17] Jnutella.org
http://www.jnutella.org/

[18] Napster.com http://www.napster.com/

[19] Winny におけるノード評価モデル
http://www.sfc.wide.ad.jp/kg/neco/blog/archives/000013.html

[20] pathchar
http://www.caida.org/tools/utilities/others/pathchar/

著者研究業績

講演

種別	題名	年月	掲載誌名	著者
査読付国際会議 1	「Karaoke Grid - A mode of Entertainment Grid」	2003 年 11 月	SC Global 2003	首藤一幸 宮之前謙一 飯島康一 大村和弘 船橋保弘 勝間亮 久和祐介 園田智也 関口智嗣 村岡洋一

謝辞

本修士論文は、私が早稲田大学 大学院理工学研究科 村岡洋一研究室に在籍する期間に行った研究をまとめたものです。本研究を進める過程で多くのご指導・ご支援を賜りました。
村岡先生には、多くの示唆に富む洞察を頂きました。深く感謝します。
村岡研究室諸氏には、議論を通して数々の助言を頂きました。また、研究室での生活でお世話になりました。
首藤さん、園田さんには本研究を進めるに当たり様々な示唆やご助言を頂きました。
家族には、研究に取組む私を常に影で支えて頂きました。
他にも様々な方から多大なご指導、ご支援を頂きました。深く感謝いたします。